SOO-BENCH: BENCHMARKS FOR EVALUATING THE STABILITY OF OFFLINE BLACK-BOX OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Black-box optimization aims to find the optima through building a model close to the black-box objective function based on function value evaluation. However, in many real-world tasks, such as design of molecular formulas and mechanical structures, it is perilous, costly, or even infeasible to evaluate the objective function value of an actively sampled solution. In this situation, optimization can only be conducted via utilizing offline historical data, which yields offline black-box optimization. Different from the traditional goal that is to pursue the optimal solution, this paper at first discloses that the goal of offline optimization is to stably surpass the offline dataset during optimization procedure. Although benchmarks called Design-Bench already exist in this emerging field, it can hardly evaluate the stability of offline optimization, and mainly provides real-world offline tasks and the corresponding offline datasets. To this end, this paper proposes benchmarks named SOO-Bench (i.e., Stable Offline Optimization Benchmarks) for offline black-box optimization algorithms, so as to evaluate the stability of surpassing the offline dataset under different data distributions. Along with SOO-Bench, we also propose a stability indicator to measure the degree of stability. Specifically, SOO-Bench includes various real-world offline optimization tasks and offline datasets under different data distributions, involving the fields of satellites, materials science, structural mechanics and automobile manufacturing. Empirically, baseline and state-of-the-art algorithms are tested and analyzed on SOO-Bench. Hopefully, SOO-Bench is expected to serve as a catalyst for rapid developments of more novel and stable offline optimization methods. The code is available at `https://anonymous.4open.science/r/SOO-Bench-9025`.

1 INTRODUCTION

Black-box optimization (BBO) is widely used in scientific and engineering fields, for example, hyper-parameter tuning (Snoek et al., 2012). BBO only needs function evaluation and does not require explicit function expression. To search for the optimal solution, a fundamental ingredient in model-based BBO is constructing a surrogate model to approximate the black-box objective function based on actively sampled solutions and their evaluated function values. However, in many real-world tasks, such as exploring drug molecular structures (Gaulton et al., 2012) and designing hardware mechanical structures (Yazdanbakhsh et al., 2021; Reagen et al., 2017), it is perilous, costly or even infeasible to actively sample solutions and evaluate their objective function values. In these cases, optimization can only be conducted via utilizing a limited number of offline historical data. That is to say, optimization algorithms need to make full use of offline dataset in hand to learn the surrogate model and determine the good solution. This yields offline model-based black-box optimization, abbreviated as offline optimization (Kumar & Levine, 2020; Trabucco et al., 2021).

A significant challenge in offline optimization is the narrow distribution of data within the offline dataset (Trabucco et al., 2021; Brookes et al., 2019; Fannjiang & Listgarten, 2020; Chen et al., 2022; 2023; Yu et al., 2021; Qi et al., 2022; Fu & Levine, 2021). In many real-world situations, we can only use narrow data distributions for offline optimization due to data collection strategies. For instance, when adjusting mechanical structure parameters, the offline dataset typically includes only the test data that the experimenter already possesses. However, these test data may be influenced by the subjective opinions of the experimenters and may not evenly cover the solution space.

Current research on offline optimization algorithms primarily focuses on finding the optima of a black-box function (Trabucco et al., 2021; Brookes et al., 2019; Fannjiang & Listgarten, 2020; Chen et al., 2022; 2023; Yu et al., 2021; Qi et al., 2022; Fu & Levine, 2021; Yuan et al., 2023). However, the surrogate model in offline optimization can be increasingly misled in the area uncovered by the offline dataset due to the narrow distribution issue. It means that the surrogate model could greatly overestimate the objective value of optima far from the offline dataset covered regions (Trabucco et al., 2021), and results in degradation during optimization procedure (Lu et al., 2023). Under this observation, we point out that stability is equally important as optimality for a comprehensive evaluation of offline algorithms. Herein, stability refers to the algorithm's ability to consistently surpass the offline dataset as much as possible during the optimization process without being misled by the narrow data distribution. ***To the best of our knowledge, this paper is the first work to disclose stability as one of the core objectives in offline black-box optimization.*** Moreover, in real-world scenarios, the degree of narrow distribution varies and is difficult to be estimated beforehand. Therefore, assessing an algorithm's stability and optimality under different levels of narrow distributions is also crucial. Unfortunately, existing benchmarks for offline optimization called Design-Bench (Trabucco et al., 2022) have limitations and inabilities in these aspects. The narrow distribution in Design-Bench is artificially constructed and fixed, and there is no established indicator for evaluating the stability of offline optimization algorithms. The field of offline optimization urgently requires a benchmark capable of comprehensive algorithm evaluation in stability and optimality, so as to boost the developments of stable offline optimization (SOO) approaches.

In response to the aforementioned demands, this paper further proposes benchmarks named SOO-Bench to evaluate offline optimization algorithms' stability and optimality. Specifically, SOO-Bench provides **(1)** real-world offline optimization tasks including satellites, materials science, structural mechanics and so on, **(2)** customizable narrow distribution levels to tailor the difficulty levels of offline optimization datasets, **(3)** a novel stability indicator called stability improvement (SI) to measure the stability of algorithms. Besides, SOO-Bench also introduces the constrained offline optimization problems. Empirically, baseline and state-of-the-art (SOTA) algorithms are tested and analyzed on SOO-Bench. By incorporating these features, SOO-Bench is able to provide a more comprehensive evaluation of offline optimization algorithms. The codes of SOO-Bench can be found at `https://anonymous.4open.science/r/SOO-Bench-9025`.

The subsequent sections respectively recap the related work, present the preliminaries, introduce the proposed SOO-Bench, depict the tasks and datasets in benchmarks, show the empirical analysis and finally conclude the paper.

2 Related Work

Although offline BBO has received widespread attention due to its application in real-world problems, there is currently only one comprehensive benchmark suite in this emerging field. Specifically, a benchmark platform called Design-Bench (Trabucco et al., 2022), which is unfortunately no longer maintained now, provides basic environments for offline optimization testing. It covers offline black-box optimization tasks in multiple fields, such as neural architecture search (Zoph & Le, 2017), DNA sequence design (A et al., 2016), drug discovery (Gaulton et al., 2012) and robot design (Ahn et al., 2019; Brockman et al., 2016). Although Design-Bench was launched as the first comprehensive benchmark suite, it only provides the basic interfaces for offline optimization, fails to analyze the narrow distribution of offline datasets fully, and does not provide a corresponding test environment. Besides, SDDObench (Zhong et al., 2024) evaluates streaming data-driven evolutionary algorithms, focusing on the need for standardized test suites in dynamic optimization, which distinguishes it from our approach. So it is urgent to propose a new benchmark to address the aforementioned issues.

In order to address the narrow distribution of offline datasets, various offline black-box optimization algorithms are proposed to handle it. For example, COMs (Trabucco et al., 2021) proposes a conservative model to penalize the proxy function value far from the offline dataset solution. IOM (Qi et al., 2022) adds an inertial regularization term to force the optimized model to maintain similar representations under different data distributions, thereby alleviating the model's performance degradation on the data far from the offline dataset. CbAS (Brookes et al., 2019) models the generative model and uses a variational autoencoder (Kingma & Welling, 2014) to build a model to find the optimal solution in the trust region within an acceptable uncertainty region. CL-DDEA (Huang & Gong,

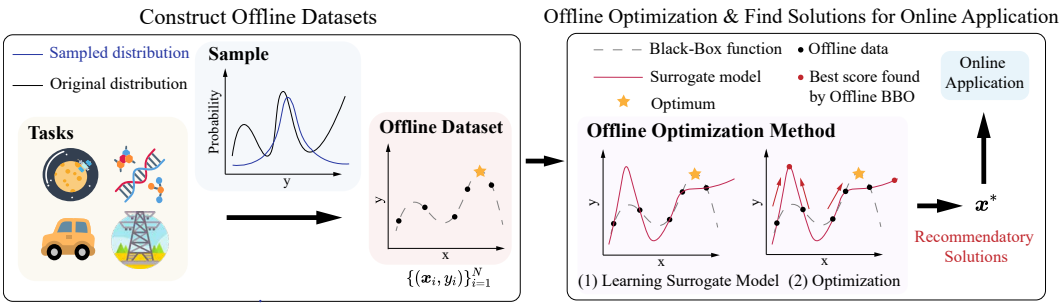


Figure 1: The process of offline black-box optimization. The left sub-figure shows the process of collecting offline data from a real black-box objective function. The right sub-figure shows the process of an offline optimization algorithm finding a satisfied solution for online application.

2022) introduces a contrastive learning model to enhance data-driven evolutionary algorithms by focusing on pairwise comparisons rather than absolute fitness values, while CC-DDEA (Gong et al., 2023) employs a hierarchical surrogate model and cooperative coevolution to address. Although relevant algorithms are designed to address the narrow distribution of offline datasets and attempt to find better solutions in areas near the offline datasets, there is currently no comprehensive benchmark suite available to test these algorithms comprehensively.

3 Offline Black-Box Optimization

3.1 Problem Statement

Let $f\colon \mathcal{X} \to \mathbb{R}$ be a black-box function, where $\mathcal{X} \subseteq \mathbb{R}^d$ is a d-dimensional solution space. In black-box optimization, the target is to find an optimal solution $\boldsymbol{x}^*$ that maximizes f, which can be written as $\boldsymbol{x}^* = \arg\max_{\boldsymbol{x}\in\mathcal{X}\subseteq\mathbb{R}^d} f(\boldsymbol{x})$. f is called objective function and $\mathcal{X}$ is called solution space. However, in the offline optimization scenario, direct interaction with the objective function is not allowed, and optimization can only be performed by accessing a static offline dataset $\mathcal{D} = \{\boldsymbol{x}_i, y_i\}_{i=1}^N$ containing N solutions and their objective values. As shown in Figure 1, most offline optimization methods use $\mathcal{D}$ to train a model $\hat{f}_{\boldsymbol{\theta}}(\boldsymbol{x})$ to fit the objective function via supervised learning: $\boldsymbol{\theta}^* \leftarrow \arg\min_{\boldsymbol{\theta}} \sum_i (\hat{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - y_i)^2$, where $\boldsymbol{\theta}$ represents the parameters of the model. Subsequently, the solution $\boldsymbol{x}_{\text{app}}$ for online optimization is found by the learned model $\hat{f}_{\boldsymbol{\theta}^*}(\boldsymbol{x})$. For example, use gradient ascent iterated T times to find the optima of the surrogate model, i.e.,

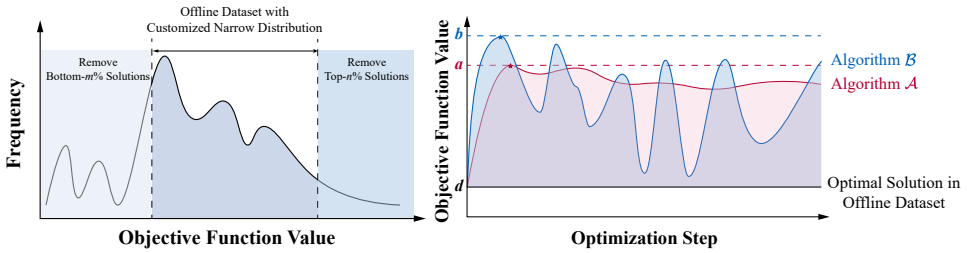
$$\boldsymbol{x}^{(t+1)} \leftarrow \boldsymbol{x}^{(t)} + \eta \nabla_{\boldsymbol{x}} \hat{f}_{\boldsymbol{\theta}^*}(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x}_t}\,,\ t = 1, 2, \ldots, T\,, \tag{1}$$

where $\boldsymbol{x}_{\text{app}} = \boldsymbol{x}_{(T)}$ is the solution output at the terminating condition set by a certain time step T for the online application. Since the exact value of T at which the optimization should stop in practical applications is unknown, and in order to avoid repeatedly adjusting the number of optimization steps for different datasets, it is important that all offline black-box optimization algorithms are designed to remain stable throughout the process. This stability ensures that the worst value of the online evaluation remains relatively favorable during the optimization process.

An offline BBO method gradually learns the characteristics of the offline dataset by training a model and then infers the online objective function. Thus, even if the offline dataset contains only local optimal solutions, the offline optimization algorithms can take advantage of the information within the dataset, identify relations among variables, especially analyze the characteristics of these local optima, and subsequently discover new solutions that surpass the best solution in the dataset.

3.2 Narrow Distribution of Offline Datasets

In offline BBO, the data distribution of offline datasets usually does not cover the entire solution space. In real-world tasks, experimenters often obtain offline datasets through expensive real experiments. For some reasons, such as experimenters' subjectivity when constructing offline datasets—where personal biases, preferences for certain parameter settings, or prior knowledge might lead to uneven



(a) An illustration of customized distribution. (b) An illustration of SI on different algorithms.

Figure 2: The illustrations of narrow data distribution issue considered and addressed in SOO-Bench and the motivation of stability indicator.

sampling—the data distribution of offline datasets cannot evenly cover all solution values in the entire solution space. For example, as shown in "Sample" of Figure 1, the black curve represents the distribution of evaluation values of solutions uniformly sampled from the solution space, while the blue curve represents the distribution of evaluation values of the offline dataset. It can be observed that the solutions in the offline dataset mainly lie in a narrow region, which results in the out-of-distribution (OOD) issue (Kumar & Levine, 2020; Trabucco et al., 2021; Brookes et al., 2019; Chen et al., 2023) studied in previous work. As a result, some areas of the solution space lack prior data information, which causes the algorithm to be unable to model these areas accurately. This narrow distribution of offline datasets presents the greatest challenge for offline optimization problems.

Therefore, when the offline BBO methods select a solution far from the offline dataset as the optimal solution in the solution space, it must exercise extreme caution. Specifically, when providing the optimal solution, the algorithm should consider the trade-off between attempting to find a better solution to surpass the optima of offline dataset and not moving too far away from the offline dataset to ensure stability. *Hence, the data distribution of the offline dataset directly determines the difficulty level of the offline optimization stability.* Controlling the distribution of the offline dataset is particularly significant for comprehensively testing the stability of offline BBO algorithms.

4 The Proposed SOO-Bench

This section introduces SOO-Bench, a more comprehensive benchmark suite to evaluate the stability of offline black-box optimization algorithms. Compared with previous work, SOO-Bench considers the customized distribution of offline datasets, allowing it to construct offline optimization tasks of different difficulties. SOO-Bench for the first time introduces the stability indicator, enabling quantitative evaluation of algorithmic stability.

4.1 Customized Narrow Distributions of Offline Datasets

The core issue of offline optimization is that the data distribution of offline datasets is usually narrow. *The narrower the data distribution covers the entire objective function search space, the more difficult the offline optimization is.* To offer a benchmark suite for comprehensive algorithmic stability and optimality performance evaluation, SOO-Bench provides offline optimization tasks with adjustable difficulty, achieved by customizing the distribution of offline datasets.

First, we provide a black-box ground-truth oracle objective function for each task. An initial dataset is obtained by uniformly sampling and evaluating the objective function. Then, the initial dataset is sorted by the value of the objective function to obtain its distribution, as shown in the curve in Figure 2 (a). Finally, datasets of different difficulty levels are constructed based on this sorted initial dataset. Specifically, the top-$n\%$ solutions by objective function value are removed to increase the difficulty of finding high-quality solutions, and the bottom-$m\%$ solutions are removed to increase the sparsity of the dataset. Through the above steps, an offline dataset with a narrow distribution in real tasks is constructed by removing solutions, as illustrated in Figure 2 (a).

4.2 The Proposed Indicator for Measuring Stability

Another problem in offline BBO is the lack of indicators to evaluate algorithm stability, which directly affects the performance of the optimal solution found by the algorithm. *We point out that the stability of an offline algorithm is reflected by its ability to continuously and stably optimize solutions that exceed the best solution in the offline dataset throughout the entire optimization process.* For example, algorithms $\mathcal{A}$ and $\mathcal{B}$ can find solutions surpassing the offline dataset's optimal solution, as shown in Figure 2 (b). However, due to the poor stability of Algorithm $\mathcal{B}$, although it found a better solution at the end of the optimization process, it failed to maintain this solution throughout the entire optimization process. In reality, when an offline BBO algorithm provides an online solution, it is often unclear which step of the optimization process should be considered. Therefore, the requirement for the algorithm to maintain a stable and high-quality solution throughout the entire optimization process becomes a critical factor affecting its effectiveness.

SOO-Bench first introduce the Stability Improvement (SI) indicator to measure algorithm stability. Specifically, for a task with an offline optimal solution $\boldsymbol{x}^*_{\mathrm{OFF}}$ (i.e., the best solution in the offline dataset), after N optimization steps, the online evaluation value of the optimal solution provided by the algorithm is denoted as $f(\boldsymbol{x}_i)$, where $i = 1, 2, \ldots, N$, and SI is then defined as follows:

$$\mathrm{SI} = \begin{cases} \frac{S - S_1}{S_2 - S_1}\,, & \text{if the algorithm can find solutions exceeding } f(\boldsymbol{x}^*_{\mathrm{OFF}})\,, \\ 0\,, & \text{if the algorithm can only find solutions same with } f(\boldsymbol{x}^*_{\mathrm{OFF}})\,, \\ -\infty\,, & \text{otherwise}\,, \end{cases} \tag{2}$$

where $S = \sum_i f(\boldsymbol{x}_i)$ represents the cumulative sum of the evaluation values curve, $S_1 = N f(\boldsymbol{x}^*_{\mathrm{OFF}})$ represents the product of the evaluation value of the optimal offline solution and optimization steps N, and $S_2 = \max_i N f(\boldsymbol{x}_i)$ represents the product of the evaluation value of the optimal solution found by the algorithm and optimization steps N.

Remark. Intuitively, as shown in Figure 2 (b), the SI indicator represents the ratio of the area under the algorithm's evaluation value curve and above the offline optimal solution to the area between the offline optimal solution and the best value found by the algorithm. When the evaluation values remain relatively stable and approach the optimal value that the algorithm can find, the SI value will be closer to 1, indicating better algorithm stability. Conversely, if the evaluation values fluctuate significantly or deviate greatly from the optimal value, the SI value will be lower or even negative. A special case occurs when the algorithm fails to find any solution that surpasses the offline optimal value. In this case, the algorithm is ineffective for offline optimization, making discussion on stability meaningless. In this scenario, we define the SI as negative infinity, as shown in Equation (2).

4.3 Extensive Tasks, Datasets and Baselines for Evaluating Stability

SOO-Bench provides an extensive real-world benchmark environment, and all task details are introduced in Section 5. Besides, SOO-Bench reproduces a variety of baseline algorithms and tests their performance through experiments, which are introduced in Section 6. It is worth noting that our benchmark suite is the first to provide constrained offline optimization tasks. We provide users with more customizable APIs. In addition to controlling task difficulty by adjusting the distribution of offline datasets, users can also create noisy offline datasets to test algorithm performance specifically.

5 Tasks and Datasets Description

This section presents the details of all tasks and datasets. An overview is shown in Table 1. Despite the tasks vary a lot, we provide a unified API on our datasets. Each task in SOO-Bench provides a ground-truth oracle objective function $f(\boldsymbol{x})$. In addition to offering basic offline datasets, it also includes an API for constructing offline datasets, allowing users to create datasets with different difficulties. We provide a detailed description of the tasks in the four benchmarks in Appendix A, which is placed in the supplementary material. Our benchmarks and example implementations are available at `https://anonymous.4open.science/r/SOO-Bench-9025`.

GTOPX: Space Mission Optimization (Schlueter et al., 2021). This paper addresses a set of real-world space mission trajectory problems (European Space Agency and Advanced Concepts Team, 2020), designed as numerical black-box optimization problems. The objective is to minimize

Table 1: An overview of the tasks in the proposed SOO-Bench. SOO-Bench includes several tasks, which are divided into four benchmarks. These tasks have discrete and continuous design spaces, and constrained and unconstrained situations to suit offline black-box optimization. N_{var} indicates the number of variables and N_{con} means the number of constraints. The symbol "/" means that the global optimal function value of the real-world black-box function is unknown.

Benchmarks	Tasks		N_{var}	N_{con}	Optimum
GTOPX	gtopx 1	Cassini 1	6	4	4.9307
	gtopx 2	Cassini 2	22	0	8.383
	gtopx 3	Messenger (reduced)	18	0	8.6299
	gtopx 4	Messenger (full)	26	0	1.9579
	gtopx 5	GTOC 1	8	6	-1581950
	gtopx 6	Rosetta	22	0	-1.3433
	gtopx 7	Cassini 1-MINLP	10	4	3.5007
CEC	cec 1	Optimal operation	7	14	-4.52912
	cec 2	Process flow	3	3	1.07654
	cec 3	Process synthesis	2	2	2
	cec 4	Three-bar	2	3	2.63896
	cec 5	Welded beam	4	5	1.67022
HYBRID	hybrid 0	Constraint task	115	2	/
	hybrid 1	Unconstraint task	115	0	/
PROTEIN	protein 1	TF Bind 8	8	0	/
	protein 2	TF Bind 10	10	0	/

the total velocity variation during interplanetary space missions. Our benchmark consists of seven GTOPX tasks. Satellite missions are particularly challenging, especially when the oracle evaluation encounters an invalid value, which returns as NaN, we indicate it as the worst value in the offline dataset. If the solution found by the offline optimization algorithm results in such a value, the true objective function value is poor, and the algorithm proves to be very unstable. The license for this dataset is GNU General Public License.

CEC Task: Industrial and Design Optimization (Kumar et al., 2020). We selected five real-world constrained problems, which are maximization tasks with constraints that are feasible when they are greater than or equal to zero. Constrained tasks are crucial in real life. When normalizing or denormalizing these tasks, the solutions found may exceed the variable boundaries. If some methods are applied to set bounds, the solutions may be suboptimal. Even if the algorithm demonstrates good stability, if the optimization results are poor, the overall outcome is still considered unsatisfactory. We aim for the offline optimization algorithm to find high-quality solutions while maintaining stability throughout the optimization process. The license for this dataset is CC-BY 4.0 License.

HYBRID: Vehicle Calibration Optimization. The Offline Hybrid Vehicle Calibration Optimization task involves developing a control strategy for hybrid vehicles to minimize fuel consumption. We utilize real-world test data to construct a simulation environment where control strategies are deployed. This environment outputs fuel consumption, battery residual energy, and mode switching counts after running a virtual vehicle over a test track. The evaluation metric is fuel consumption, with constraints on battery residual energy and mode switching counts. Each control strategy is uniquely defined by 115 parameters, and the evaluation environment generates and tests these strategies to assess performance and constraint adherence. We provide two types of tasks: constrained and unconstrained. Unconstrained tasks are implemented by adding constraints to the objective function. For high-dimensional optimization tasks, the search space is vast. When the offline optimization algorithm overestimates, it can lead to significant deviations, resulting in unstable optimization performance. The license for this dataset is CC-BY 4.0 License.

PROTEIN: DNA Sequence Optimization (Trabucco et al., 2022). The variable space comprises sequences of four categorical variables. The objective of the two tasks is to identify the optimal 8-nucleotide DNA sequence that exhibits the highest binding affinity to a specific transcription factor.

Discrete tasks play a crucial role in offline optimization. However, in these tasks, offline optimization may identify non-existent molecules. The license for this dataset is MIT License.

6 EXPERIMENT

To simulate a more realistic data distribution, we choose a dataset size that is 1000 times the variable dimension. At the same time, to further simulate the narrow distribution, the missing $m\%$ near the worst value and the missing $n\%$ near the optimal value are used to ensure that the data volume is small and missing near the optimal value, as shown in Figure 2 (a). In this paper, we select the middle 50% of the data (i.e., $m\% - n\% = 50\%$) to construct a simulated dataset as a reasonable baseline without leveraging any prior knowledge. Since the proposed benchmark is highly flexible and customizable, it enables users to modify the data volume, $m\%$ and $n\%$ as needed. At the same time, we designed an experiment with gradual missing data near the optimal value, aiming to construct datasets of different difficulty levels, see the Appendix C for details. Through these settings, we hope to systematically analyze and evaluate the impact of different data distributions on the performance of different offline optimization algorithms.

6.1 OFFLINE OPTIMIZATION ALGORITHMS

We test a range of baseline and SOTA offline optimization algorithms on each of task. Specifically, we compare with two categories of algorithms: (1) those that address unconstrained problems, including classic baselines: BO-qEI (Wilson et al., 2017), CMA-ES (Hansen, 2006), and Offline BBO methods: autofocusing CbAS (Fannjiang & Listgarten, 2020), TTDDEA (Huang et al., 2021), ARCOO (Lu et al., 2023), Tri-mentoring (Chen et al., 2023), and (2) those that address constrained problems, including CARCOO, CCOMs, DDEA-PF, DDEA-SPF (Huang & Wang, 2021a). Since classical methods lack specific methodologies for offline optimization problems, they cannot be directly applied to such problems. Therefore, we introduce a guided training agent model to provide optimization guidance through agent prediction, thereby indirectly solving the offline optimization problem. In this section, we briefly discuss these algorithms and evaluate them in the next section. Due to page limitation, the implementation codes links are listed in Appendix B.

BO-qEI: Offline Bayesian optimization is performed to minimize a learned surrogate function $\hat{f}(\boldsymbol{x})$ by training an ensemble of neural network models. Candidate solutions were generated with a Gaussian Process model and labeled with values from $\hat{f}(\boldsymbol{x})$. We employed the quasi-expected Improvement (qEI) acquisition function within the BoTorch framework (Balandat et al., 2019). **CMA-ES:** We compute the value of learned surrogate function $\hat{f}(\boldsymbol{x})$ on the samples $\boldsymbol{x}_t$, which is obtained from the distribution $\mathcal{N}(\mu_t, \Sigma_t)$ at an iteration t. We then adapted the covariance matrix to refine the belief distribution, repeating this process multiple times. **Autofocusing CbAS:** Autofocusing CbAS learns a density model $p_0(\boldsymbol{x})$ of $\boldsymbol{x}$ to approximate the data distribution, then gradually use importance sampling to re-training $\hat{f}(\boldsymbol{x})$ under current distribution $p_t(\boldsymbol{x})$, and adapts it towards the optimized solution. **ARCOO:** ARCOO constructs the surrogate model $\hat{f}(\boldsymbol{x})$ combined with the energy model, which is used to characterize the risk of degradation. After construction, a risk suppression factor is applied to control the risk. **Tri-mentoring:** This approach constructs three surrogate models and uses ranking supervision signals for mutual mentoring. After that, adaptive soft-labeling to learn more accurate labels. **TTDDEA:** By dividing the offline dataset into three equal parts randomly, we build three surrogate models respectively. We selected a high-confidence pseudo-label to fill in each other's datasets, then retrained the surrogate model, generated offspring, and repeated this multiple times. **CCDDEA:** CCDDEA designs a hierarchical surrogate-joint learning model to be able to guide the evolving population to search at different granularities, and then optimizes at the global and local subspace levels in a cooperatively coordinated evolutionary manner. **CARCOO:** This is a simple constrained version of ARCOO. We incorporate the degree of constraint violation into the risk assessment and consider solutions that violate the constraints as high risk. **CCOMs:** This is a reproduction version of the PRIME (Kumar et al., 2022) method. Since we did not find its codes, we simply implemented PRIME according to the paper, that is, making the objective function value that violates the constraint as bad as possible. **DDEA-PF & DDEA-SPF:** It combines common constraint processing techniques with offline data-driven evolutionary algorithms to handle constrained optimization problems. This is achieved by constructing proxy models for constraints

Table 2: Overall results in GTOPX unconstrianed scenario. Results are averaged over five times, and "±" indicates the standard deviation. $f(\boldsymbol{x}^*_{\text{OFF}})$ means the optimal objective function value in the offline dataset. FS (i.e., final score) means the function value that an offline optimization algorithm finds in the final step during optimization process. FS measures optimality while SI measures stability.

Tasks	GTOPX 2		GTOPX 3		GTOPX 4		GTOPX 6	
$f(\boldsymbol{x}^*_{\text{OFF}})$	196.21		151.68		216.34		112.11	
Metrics	FS ↓	SI ↑	FS ↓	SI ↑	FS ↓	SI ↑	FS ↓	SI ↑
BO	95.45±18.43	0.69±0.02	75.78±29.38	0.7±0.03	117.39±17.11	0.68±0.04	58.11±8.25	0.63±0.04
CMAES	196.21±1.18	0.00±0.00	151.68±0.53	0.00±0.00	216.34±0.61	0.00±0.00	112.11±0.38	0.00±0.00
CBAS	196.21±1.18	0.00±0.00	86.51±3.97	0.71±0.06	208.8±15.50	0.14±0.02	87.08±31.05	0.34±0.09
TTDDEA	224.17±53.87	$-\infty$	156.92±91.05	$-\infty$	260.4±54.89	$-\infty$	148.76±50.67	$-\infty$
ARCOO	90.73±10.98	0.78±0.05	65.88±13.12	0.85±0.01	102.84±21.76	0.79±0.04	65.17±13.30	0.74±0.08
Tri-mentoring	129.47±54.75	$-\infty$	140.23±22.88	$-\infty$	176.31±37.30	0.86±0.18	112.11±0.38	$-\infty$
CCDDEA	197.25±3.70	$-\infty$	152.71±4.01	$-\infty$	216.18±3.29	$-\infty$	112.46±1.88	$-\infty$

Table 3: Overall results in GTOPX constrianed scenario. Details are the same as Table 2. The symbol "-" means that the algorithm cannot work because of too few solutions that satisfly the constraints.

Tasks	GTOPX 1		GTOPX 5		GTOPX 7	
$f(\boldsymbol{x}^*_{\text{OFF}})$	75.3		5.41		346.2	
Metrics	FS ↓	SI ↑	FS ↓	SI ↑	FS ↓	SI ↑
CARCOO	72.97±5.64	0.34±0.04	1.08±5.85	$-\infty$	189.14±118.59	0.64±0.06
CCOMs	68.61±108.20	$-\infty$	–	$-\infty$	307.81±19.07	0.99±0.00
DDEA-PF	282.57±35.59	$-\infty$	0.00±0.00	$-\infty$	953.59±27.33	$-\infty$
DDEA-SPF	282.57±35.59	$-\infty$	0.00±0.00	$-\infty$	953.59±27.33	$-\infty$

and objective functions respectively, and using constraint processing techniques to process constraint proxy models and objective proxy models.

6.2 Results and Analysis

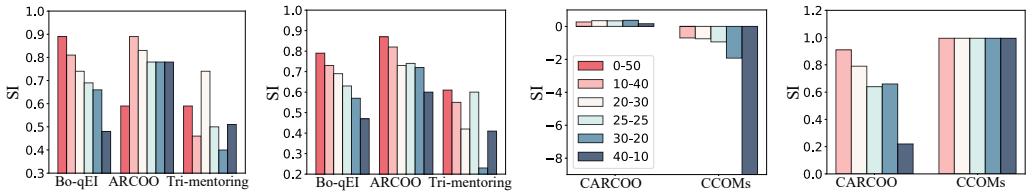
This scenario represents normal settings in real-world conditions where the maximum and minimum values of the objective values are proportionally missing. In the unconstrained optimization case, we have four tasks in GTOPX. As shown in Table 2, we can obtain two key observations:

• Baselines like BO-qEI are competitive with offline BBO methods in both FS and SI, but CMA-ES performs differently. This is in contrast to Design-Bench, which might be due to the lower optimization dimension in GTOPX, and that the objective function is non-smooth, making it unsuitable for methods like CMA-ES that rely on guidance through a covariance matrix.

• Existing offline BBO methods (e.g., CBAS, ARCOO, Tri-mentoring) perform well across different benchmarks, except for TTDDEA. This is because TTDDEA, being an evolutionary-based method, relies on high-confidence data augmentation without considering the narrow distribution problem. Notably, ARCOO achieves the best stability performance by evaluating the risk of generated solutions during optimization through a learned energy model, thereby controlling the step size during gradient ascent. This highlights the significant role of risk control in the stability of optimization results.

In the constrained optimization case, we have three tasks in GTOPX. As shown in Table 3, we can obtain two key observations:

• Experimental results show that simply constructed offline BBO methods for handling constraint problems are successful on both unconstrained and constrained tasks. However, we find that DE-PF and DE-SPF perform poorly on satellite missions. This result suggests that directly combining online constraint handling techniques with offline evolutionary methods may have difficulty capturing features in complex tasks with sufficient accuracy. In addition, inaccurate surrogate models may incorrectly guide online constraint handling techniques, resulting in poor performance on satellite missions or even failure to find solutions that meet the constraints.



(a) gtopx 2 (b) gtopx 6 (c) gtopx 1 (d) gtopx 8

Figure 3: Stability under different $n\%$ and $m\%$. Unconstrained: (a), (b). Constrained: (c), (d).

• We find that simply modifying the previous unconstrained offline optimization method can also achieve good optimization results and stability on complex satellite missions, which shows that the previous unconstrained offline optimization method is very competitive.

Optimization Results under Different $n\%$ and $m\%$. In real-world scenarios, we often do not know the distribution of the dataset. We simulate this by keeping the total missing proportion constant while varying n% and m%. Based on the results in Table 2 and Table 3, we selected BO-qEI, ARCOO, and Tri-mentoring for unconstrained benchmarks and CARCOO and CCOMs for constrained benchmarks, as they achieve satisfactory optimization stability in balanced scenarios. As shown in Figure 3 (a) and Figure 3 (b), the optimization stability of BO-qEI gradually decreases as n% increases. In contrast, ARCOO initially even shows some improvement but generally remains stable. Tri-mentoring performs best at $n = 20\%, m = 30\%$, but its performance is subpar at other times. In unconstrained benchmarks, we can obtain two observations:

• Classical baselines like BO-qEI, which originate from online optimization scenarios, are highly sensitive to data missing in the optimal search region. In contrast, offline BBO methods are relatively stable, demonstrating their advantage in different narrow distributions and validating their effectiveness in real-world applications.

• ARCOO outperforms Tri-mentoring in various $n\%$ and $m\%$ situations, indicating that evaluating the risk in the optimization process significantly contributes to optimization stability across different narrow distributions.

• Although CCOMs are relatively stable in some cases, they often find poor solutions and stagnate, indicated by a relatively high SI and sub-optimal optimization results. The reason may be that the constraints are too strict, making the algorithm less inclined to explore outside current solution space.

• On the gtopx 1 task, the metrics of CCOMs are all below 0. This is because although CCOMs can find better values than the optimal objective value in offline dataset, their subsequent performance directly declines and is worse than the offline optimal objective value, resulting in poor SI values.

7 Conclusion and Discussion

This paper proposes SOO-Bench, a benchmark suite to promote stable offline black-box optimization. SOO-Bench provides a variety of real offline optimization tasks. Considering that the narrow distribution of offline datasets is a vital challenge of offline optimization, SOO-Bench provides APIs that can customize the distribution of offline datasets to construct tasks with different difficulties, thereby comprehensively testing the performance of algorithms. Notably, SOO-Bench for the first time reveals the significant of stability of offline BBO and introduces an indicator called Stability Improvement (SI) to quantify the stability. Finally, SOO-Bench reproduces offline optimization baselines and conducts experiments to evaluate algorithmic stability and optimality.

There are still some limitations of SOO-Bench. First, the variety of tasks we provide is still not rich enough, and there are many unexplored black-box optimization scenarios (e.g., robotic control). Second, we emphasize that optimization stability is crucial in the offline optimization, the proposed metric can trade off between the optimality of the output solution and the stability of optimization process, but we believe that other reasonable metrics exist. The future work will explore real-world scenarios by providing new datasets and tasks, and investigate a range of data size percentages (i.e., different degree about the missing $m\%$ near the worst value and the missing $n\%$ near the optimal

value) to conduct a more comprehensive analysis. Meanwhile, we will further improve SOO-Bench, develop more realistic narrow distribution methods to simulate real offline datasets, and propose a comprehensive evaluation system containing richer evaluation indicators.

8 Ethics and Reproducibility Statements

Ethics. This paper does not involve human subjects, personal data, or sensitive information. All datasets used for testing are publicly available, and no proprietary or confidential information has been utilized. We take responsibility for any potential violation of rights and for ensuring compliance with data licensing requirements.

Reproducibility. Experimental settings are described in Section 6.1 with further details of the methods included in Appendix A-E. The datasets utilized in this paper are all publicly available and open-source. The link to our anonymized repository that includes codes, datasets, documents, demo and license can be found from `https://anonymous.4open.science/r/SOO-Bench-9025`.

References

Barrera Luis A, Vedenko Anastasia, Kurland Jesse V, Rogers Julia M, Gisselbrecht Stephen S, Rossin Elizabeth J, Woodard Jaie, Mariani Luca, Kock Kian Hong, and Inukai Sachi. Survey of variation in human transcription factors reveals prevalent dna binding changes. *Science*, 351(6280): 1450–1454, 2016.

Michael Ahn, Henry Zhu, Kristian Hartikainen, Hugo Ponte, Abhishek Gupta, Sergey Levine, and Vikash Kumar. ROBEL: robotics benchmarks for learning with low-cost robots. In *Proceedings of the 3rd Annual Conference on Robot Learning*, pp. 1300–1313, Osaka, Japan, 2019.

Maximilian Balandat, Brian Karrer, Daniel R Jiang, Samuel Daulton, Benjamin Letham, Andrew Gordon Wilson, and Eytan Bakshy. Botorch: Programmable Bayesian optimization in pytorch. *arXiv preprint arXiv:1910.06403*, 117, 2019.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *CoRR*, abs/1606.01540, 2016.

David H. Brookes, Hahnbeom Park, and Jennifer Listgarten. Conditioning by adaptive sampling for robust design. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 773–782, Long Beach, CA, 2019.

Can Chen, Yingxue Zhang, Jie Fu, Xue (Steve) Liu, and Mark Coates. Bidirectional learning for offline infinite-width model-based optimization. In *Advances in Neural Information Processing Systems 35*, pp. 29454–29467, New Orleans, LA, 2022.

Can Chen, Christopher Beckham, Zixuan Liu, Xue (Steve) Liu, and Chris Pal. Parallel-mentoring for offline model-based optimization. In *Advances in Neural Information Processing Systems 36*, New Orleans, LA, 2023.

European Space Agency and Advanced Concepts Team. GTOP Database - global optimisation trajectory problems and solutions. `https://www.esa.int/gsp/ACT/projects/gtop/`, 2020. Archived webpage.

Clara Fannjiang and Jennifer Listgarten. Autofocused oracles for model-based design. In *Advances in Neural Information Processing Systems 33*, pp. 12945–12956, Virtual Event, 2020.

Justin Fu and Sergey Levine. Offline model-based optimization via normalized maximum likelihood estimation. In *Proceedings of the 9th International Conference on Learning Representations*, Virtual Event, 2021.

Anna Gaulton, Louisa J. Bellis, A. Patrícia Bento, Jon Chambers, Mark Davies, Anne Hersey, Yvonne Light, Shaun McGlinchey, David Michalovich, Bissan Al-Lazikani, and John P. Overington. ChEMBL: A large-scale bioactivity database for drug discovery. *Nucleic Acids Research*, 40: 1100–1107, 2012.

Yue-Jiao Gong, Yuan-Ting Zhong, and Hao-Gan Huang. Offline data-driven optimization at scale: A cooperative coevolutionary approach. *IEEE Transactions on Evolutionary Computation*, 2023.

Nikolaus Hansen. The CMA evolution strategy: A comparing review. In *Towards a New Evolutionary Computation - Advances in the Estimation of Distribution Algorithms*, volume 192, pp. 75–102. 2006.

Hao-Gan Huang and Yue-Jiao Gong. Contrastive learning: An alternative surrogate for offline data-driven evolutionary computation. *IEEE Transactions on Evolutionary Computation*, 27(2): 370–384, 2022.

Pengfei Huang and Handing Wang. Comparative empirical study on constraint handling in offline data-driven evolutionary optimization. *Applied Soft Computing*, 110:107603, 2021a.

Pengfei Huang and Handing Wang. Comparative empirical study on constraint handling in offline data-driven evolutionary optimization. *Applied Soft Computing*, 110:107603, 2021b.

Pengfei Huang, Handing Wang, and Yaochu Jin. Offline data-driven evolutionary optimization based on tri-training. *Swarm and Evolutionary Computation*, 60:100800, 2021.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *Proceedings of the 2nd International Conference on Learning Representations*, Banff, Canada, 2014.

Abhishek Kumar, Guohua Wu, Mostafa Z. Ali, Rammohan Mallipeddi, Ponnuthurai Nagaratnam Suganthan, and Swagatam Das. A test-suite of non-convex constrained optimization problems from the real-world and some baseline results. *Swarm and Evolutionary Computation*, 56:100693, 2020.

Aviral Kumar and Sergey Levine. Model inversion networks for model-based optimization. In *Advances in Neural Information Processing Systems 33*, pp. 5126–5137, Virtual Event, 2020.

Aviral Kumar, Amir Yazdanbakhsh, Milad Hashemi, Kevin Swersky, and Sergey Levine. Data-driven offline optimization for architecting hardware accelerators. In *Proceedings of the 10th International Conference on Learning Representations*, Virtual Event, 2022.

Huakang Lu, Hong Qian, Yupeng Wu, Ziqi Liu, Ya-Lin Zhang, Aimin Zhou, and Yang Yu. Degradation-resistant offline optimization via accumulative risk control. In *Proceedings of the 26th European Conference on Artificial Intelligence*, pp. 1609–1616, Kraków, Poland, 2023.

Han Qi, Yi Su, Aviral Kumar, and Sergey Levine. Data-driven offline decision-making via invariant representation learning. In *Advances in Neural Information Processing Systems 35*, pp. 13226–13237, New Orleans, LA, 2022.

Brandon Reagen, José Miguel Hernández-Lobato, Robert Adolf, Michael A. Gelbart, Paul N. Whatmough, Gu-Yeon Wei, and David M. Brooks. A case for efficient accelerator design space exploration via Bayesian optimization. In *IEEE/ACM International Symposium on Low Power Electronics and Design*, pp. 1–6, 2017.

Martin Schlueter, Mehdi Neshat, Mohamed Wahib, Masaharu Munetomo, and Markus Wagner. GTOPX space mission benchmarks. *SoftwareX*, 14:100666, 2021.

Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems 25*, pp. 2960–2968, Lake Tahoe, NV, 2012.

Brandon Trabucco, Aviral Kumar, Xinyang Geng, and Sergey Levine. Conservative objective models for effective offline model-based optimization. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 10358–10368, Virtual Event, 2021.

Brandon Trabucco, Xinyang Geng, Aviral Kumar, and Sergey Levine. Design-Bench: Benchmarks for data-driven offline model-based optimization. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 21658–21676, Baltimore, MD, 2022.

James T. Wilson, Riccardo Moriconi, Frank Hutter, and Marc Peter Deisenroth. The reparameterization trick for acquisition functions. *CoRR*, abs/1712.00424, 2017.

Amir Yazdanbakhsh, Christof Angermüller, Berkin Akin, Yanqi Zhou, Albin Jones, Milad Hashemi, Kevin Swersky, Satrajit Chatterjee, Ravi Narayanaswami, and James Laudon. Apollo: Transferable architecture exploration. *CoRR*, abs/2102.01723, 2021.

Sihyun Yu, Sungsoo Ahn, Le Song, and Jinwoo Shin. RoMA: Robust model adaptation for offline model-based optimization. In *Advances in Neural Information Processing Systems 34*, pp. 4619–4631, Virtual Event, 2021.

Ye Yuan, Can Chen, Zixuan Liu, Willie Neiswanger, and Xue (Steve) Liu. Importance-aware co-teaching for offline model-based optimization. In *Advances in Neural Information Processing Systems 36*, New Orleans, LA, 2023.

Yuanting Zhong, Xincan Wang, Yuhong Sun, and Yue-Jiao Gong. Sddobench: A benchmark for streaming data-driven optimization with concept drift. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 59–67, 2024.

Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning. In *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France, 2017.

APPENDIX

This appendix is structured into four sections to improve understanding and facilitate the reproduction of our benchmark research. Task Description Section elaborates on the objective function, constraints, and variables used in our study. Experimental Details Section provides comprehensive information on the methodologies, code implementations, and resource usage of all methods compared. Experimental Results Section thoroughly discusses the research results and findings in detail. Finally, Hyperparameter Analysis Section itemizes the specific settings and configurations utilized in the experiments. Our benchmark and comparison algorithms are available at `https://anonymous.4open.science/r/SOO-Bench-9025`.

A TASK DESCRIPTION

In this section, we detail task description in SOO-Bench. Specifically, we answer **(1)** what is the goal and background of each task? **(2)** where are the tasks from? **(3)** what do the variables and constraints in each task represent?

A.1 GTOPX: SPACE MISSION BENCHMARKS

For those interested in the task, more information can be found on the official website for the benchmark. To access detailed descriptions, methodologies, and data related to the task, please visit the following URL: `http://www.midaco-solver.com/index.php/about/benchmarks/gtopx`. This page provides essential resources for understanding the scope and requirements of the benchmark challenges.

A.1.1 GTOPX 1: CASSINI 1

The Cassini 1 benchmark simulates an interplanetary mission targeting Saturn, with the mission's aim being to enter a specific orbit around Saturn characterized by an intracentric radius of 108,950 km and an eccentricity of 0.98. The primary goal of this benchmark is to minimize the total ΔV, or change in velocity, required throughout the mission, which includes the launch and orbital insertion maneuvers. This scenario utilizes six decision variables and incorporates four constraints that set maximum limits on the proximity of the center during four flyby maneuvers. The variables' descriptions of this task include var_1 and var_2.

A.1.2 GTOPX 2: CASSINI 2

The Cassini 2 benchmark, which models complex interplanetary missions to Saturn, including critical maneuvers like the Deep Space Maneuver (DSM), presents a more challenging scenario than the gtopx 1 benchmark. Unlike Cassini 1, which focuses on orbital insertion, the primary objective of this benchmark is to achieve a rendezvous with Saturn, aiming to minimize the total ΔV required throughout the mission. This benchmark involves handling 22 decision variables. The variables' descriptions of this task include var_1, var_2, var_3, var_4, var_5, var_6, and var_7.

A.1.3 GTOPX 3: MESSENGER (REDUCED)

The third benchmark, known as Messenger (reduced), is a simulation of interplanetary missions to Mercury, excluding any resonant flybys of the planet. The main objective of this benchmark is to minimize the total ΔV over the course of the mission. It involves 18 decision variables and the explanation of the variables is given in Table 4.

A.1.4 GTOPX 4: MESSENGER (FULL)

The fourth benchmark, titled Messenger (full), models interplanetary missions to Mercury, incorporating resonant flybys of the planet. This benchmark aims to minimize the total ΔV incurred throughout the mission. It features 26 decision variables, which includes all variables in Table 4.

Table 4: Description of variables in GTOPX tasks (Schlueter et al., 2021).

Variable	Descriptions	gtopx 1	gtopx 2
var_1	Initial day measured from 1-Jan 2000	✓	✓
var_2	Time interval between events (e.g. departure, fly-by, capture)	✓	✓
var_3	Initial excess hyperbolic speed (km/S)	×	✓
var_4	Angles of excess velocity in a hyperbolic trajectory	×	✓
var_5	Fraction of the time interval after which DSM occurs	×	✓
var_6	Radius of flyby (in planet radii)	×	✓
var_7	Orientation of the trajectory angle in the planet's B-plane approach vector	×	✓

A.1.5 GTOPX 5: GTOC1

The fifth benchmark, known as GTOC1, models a complex space mission involving multi-gravity assists to asteroid TW229. The objective of this mission is to maximize the change in the semi-major axis of the asteroid's orbit. This benchmark incorporates 8 decision variables and includes 4 constraints that set limits on the proximity to the center during each of the four flyby maneuvers. The variables description includes var_1 and var_2.

A.1.6 GTOPX 6: ROSETTA

The sixth benchmark, named Rosetta, emulates multi-gravity-assisted space missions to Comet 67P/Churyumov-Gerasimenko, including the execution of a DSM. The primary objective of this benchmark is to minimize the total ΔV required throughout the mission. It encompasses 22 decision variables and description of variables are shown in Table 4.

A.1.7 GTOPX 7: CASSINI1-MINLP

The last benchmark, cassini1-minLP, is a mixed integer extension of a Cassini1 instance. While in the original Cassini1 instance, the order of planetary flybys was fixed as Venus-Venus-Earth-Jupiter, Cassini1-MINLP treated all four flybys as discrete decision variables. Every planet in the solar system (plus the dwarf planet Pluto) is a viable option to fly by any of the four planets. The benchmark involves 12 decision variables and further considers four constraints. Descriptions of all variables are in Table 4.

A.2 CEC TASK: INDUSTRIAL AND DESIGN PROBLEMS

For detailed information and access to the resources associated with the task, please visit the provided URL: `https://github.com/P-N-Suganthan/2020-RW-Constrained-Optimization`. This link leads to the GitHub repository where you can find all the necessary files, including source code and documentation, to understand and engage with the benchmark suite effectively.

A.2.1 CEC 1: OPTIMAL OPERATION OF ALKYLATION UNIT

The initial benchmark test is termed "Optimal Operation of Alkylation Unit". This test focuses on optimizing the octane number of the olefin feed in an acidic environment, with the main goal to enhance the alkylating product. The benchmark involves 7 decision variables and incorporates 14 constraints, which are designed to limit onboard fuel and launcher performance. The problem can be formulated as follows.

Minimize:
$f(\bar{x}) = -(0.035x_1x_6 + 1.715x_1 + 10.0x_2 + 4.0565x_3 - 0.063x_3x_5)$
subject to:
$$
\begin{aligned}
&g_1(\bar{x}) = 0.0059553571x_6^2x_1 + 0.88392857x_3 - 0.1175625x_6x_1 - x_1 \leq 0,\\
&g_2(\bar{x}) = 1.1088x_1 + 0.1303533x_1x_6 - 0.0066033x_1x_6^2 - x_3 \leq 0,\\
&g_3(\bar{x}) = 6.66173269x_6^2 - 56.596669x_4 + 172.39878x_5 - 10000 - 191.20592x_6 \leq 0\\
&g_4(\bar{x}) = 1.08702x_6 - 0.03762x_6^2 + 0.32175x_4 + 56.85075 - x_5 \leq 0,\\
&g_5(\bar{x}) = 0.006198x_7x_4x_3 + 2462.3121x_225.125634x_2x_4 - x_3x_4 \leq 0,\\
&g_6(\bar{x}) = 161.18996x_3x_4 + 5000.0x_2x_4 - 489510.0x_2 - x_3x_4x_7 \leq 0,\\
&g_7(\bar{x}) = 0.33x_7 + 44.333333 - x_5 \leq 0,\\
&g_8(\bar{x}) = 0.022556x_5 - 1.0 - 0.007595x_7 \leq 0\\
&g_9(\bar{x}) = 0.00061x_3 - 1.0 - 0.0005x_1 \leq 0,\\
&g_{10}(\bar{x}) = 0.819672x_1 - x_3 + 0.819672 \leq 0,\\
&g_{11}(\bar{x}) = 24599.9x_2250x_2x_4 - x_3x_4 \leq 0,\\
&g_{12}(\bar{x}) = 1020.4082x_4x_2 + 1.2244898x_3x_4 - 100000x_2 \leq 0,\\
&g_{13}(\bar{x}) = 6.25x_1x_6 + 6.25x_1 - 7.625x_3 - 100000 \leq 0,\\
&g_{14}(\bar{x}) = 1.22x_3 - x_6x_1 - x_1 + 1.0 \leq 0.
\end{aligned}
\tag{3}
$$
with bounds:
$1000 \leq x_1 \leq 2000, 0 \leq x_2 \leq 100,$
$2000 \leq x_3 \leq 4000, 0 \leq x_4 \leq 100,$
$0 \leq x_5 \leq 100, 0 \leq x_6 \leq 20,$
$0 \leq x_7 \leq 200.$

A.2.2 CEC 2: PROCESS FLOW SHEETING PROBLEM

The second benchmark is characterized as a non-convex constrained optimization problem. This benchmark incorporates three decision variables and is governed by three constraints. It is noted for having an optimal objective function value of $f(x) = 1.07654$. The formulation of this problem can be shown as follows.

Minimize:
$$
\begin{aligned}
&f(\bar{x}) = -0.7x_3 + 0.8 + 5\left(0.5 - x_1\right)^2\\
&\text{subject to:}\\
&g_1(\bar{x}) = -\exp\left(x_1 - 0.2\right) - x_2 \leq 0,\\
&g_2(\bar{x}) = x_2 + 1.1x_3 \leq -1.0,\\
&g_3(\bar{x}) = x_1 - x_3 \leq 0.2.\\
&\text{with bounds:}\\
&2.22554 \leq x_2 \leq -1, 0.2 \leq x_1 \leq 1, x_3 \in \{0, 1\}.
\end{aligned}
\tag{4}
$$

A.2.3 CEC 3: PROCESS SYNTHESIS PROBLEM

The third benchmark, referred to as the Process Synthesis Problem, is defined as a non-convex constrained optimization problem. This benchmark features two decision variables and is subject to two constraints. It is recognized for achieving an best objective function value of $f(x) = 2.0$. The

problem can be defined as follows.

$$
\begin{aligned}
&\text{Minimize:}\\
&f(\bar{x}) = x_2 + 2x_1\\
&\text{subject to:}\\
&g_1(\bar{x}) = -x_1^2 - x_2 + 1.25 \leq 0,\\
&g_2(\bar{x}) = x_1 + x_2 \leq 1.6.\\
&\text{with bounds:}\\
&0 \leq x_1 \leq 1.6\,,\\
&x_2 \in \{0, 1\}\,.
\end{aligned}
\tag{5}
$$

A.2.4 CEC 4: THREE-BAR TRUSS DESIGN PROBLEM

The fourth benchmark, known as the Three-bar Truss Design Problem, originates from the field of civil engineering and involves a complex constrainted setup. The primary objective of this problem is to reduce the weight of the truss structure. The constraints are based on the stress limits for each bar, leading to a problem characterized by a linear objective function, two decision variables, and three nonlinear constraints. This benchmark's optimal objective function value is $f(x) = 2.63896$. The problem can be defined as below.

$$
\begin{aligned}
&\text{Minimize:}\\
&f(\bar{x}) = l\left(x_2 + 2\sqrt{2}x_1\right)\\
&\text{subject to:}\\
&g_1(\bar{x}) = \frac{x_2}{2x_2x_1 + \sqrt{2}x_1^2}P - \sigma \leq 0,\\
&g_2(\bar{x}) = \frac{x_2 + \sqrt{2}x_1}{2x_2x_1 + \sqrt{2}x_1^2}P - \sigma \leq 0,\\
&g_3(\bar{x}) = \frac{1}{x_1 + \sqrt{2}x_2}P - \sigma \leq 0.\\
&\text{where,}\\
&l = 100, P = 2, \text{and } \sigma = 2\,.\\
&\text{with bounds:}\\
&0 \leq x_1, x_2 \leq 1\,.
\end{aligned}
\tag{6}
$$

A.2.5 CEC 5: WELDED BEAM DESIGN

The fifth benchmark, known as the Welded Beam Design, primarily aims to minimize the cost of constructing welded beams. This engineering challenge is defined by four variables and five constraints, which guide the development of the beam. The most acclaimed outcome of this benchmark is recorded with an best objective function value of $f(x) = 1.67022$. The description of this problem

is shown below.

$$
\begin{aligned}
&\text{Minimize:}\\
&f(\bar{x}) = 0.04811 x_3 x_4 \left(x_2 + 14\right) + 1.10471 x_1^2 x_2\\
&\text{subject to:}\\
&g_1(\bar{x}) = x_1 - x_4 \leq 0,\\
&g_2(\bar{x}) = \delta(\bar{x}) - \delta_{\max} \leq 0,\\
&g_3(\bar{x}) = P \leq P_c(\bar{x}),\\
&g_4(\bar{x}) = \tau_{\max} \geq \tau(\bar{x}),\\
&g_5(\bar{x}) = \sigma(\bar{x}) - \sigma_{\max} \leq 0.\\
&\text{where,}\\
&\tau = \sqrt{\tau'^2 + \tau''^2 + 2\tau'\tau''\frac{x_2}{2R}}, \tau'' = \frac{RM}{J}, \tau' = \frac{P}{\sqrt{2}x_2 x_1},\\
&M = p\left(\frac{x_2}{2} + L\right),\\
&R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_{xw3}}{2}\right)^2}, J = 2\left(\left(\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2\right)\sqrt{2}x_1 x_2\right),\\
&\sigma(\bar{x}) = \frac{6PL}{x_4 x_3^2}, \delta(\bar{x}) = \frac{6PL^3}{E x_3^2 X_4}, P_c(\bar{x}) = \frac{4.013 E x_3 x_4^3}{6L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)\\
&L = 14\text{in}, P = 6000\text{lb}, E = 30.10^6\text{psi}, \sigma_{\max} = 30,000\text{psi}, \tau_{\max} = 13,600\text{psi},\\
&G = 12.10^6\text{psi}, \delta_{\max} = 0.25\text{in}\\
&\text{with bounds:}\\
&0.1 \leq x_3, x_2 \leq 10\\
&0.1 \leq x_4 \leq 2\\
&0.125 \leq x_1 \leq 2\,.
\end{aligned}
\tag{7}
$$

A.3 HYBRID: VEHICLE CALIBRATION OPTIMIZATION TASK

The benchmark, known as Hybrid Vehicle Calibration, focuses on optimizing control schemes for hybrid vehicles to improve overall efficiency. This intricate engineering task integrates an electric motor with a conventional engine, necessitating precise coordination to optimize vehicle dynamics like speed and acceleration while conforming to battery capacity limits. The objective is to fine-tune these control strategies, which involve 115 distinct parameters, to ensure the engine operates at peak efficiency across different driving conditions. The assessment aims to reduce fuel usage while ensuring that battery management and mode transitions adhere to predefined constraints, leveraging a comprehensive dataset for analysis. The specific constrained and unconstrained construction methods are as follows:

$$
\text{value} = \begin{cases} \text{objective value} + \text{fuel\_punish\_mode} + \text{punish\_soc} & \text{unconstrainted task}\,, \\ \text{objective value}, \text{fuel\_punish\_mode} \geq 0, \text{punish\_soc} \geq 0 & \text{constrainted task}\,. \end{cases} \tag{8}
$$

A.4 PROTEIN: DNA SEQUENCE OPTIMIZATION

The tasks TF Bind 8 and TF Bind 10 explore the affinity of various DNA sequences for several human transcription factors. In our study, we specifically targeted the transcription factor SIX6 REF R1, with the aim to design an nucleotide sequence that exhibits high binding affinity to this factor. The datasets for TF Bind 8 and TF Bind 10 encompass all possible combinations of nucleotides for sequences of lengths 8 and 10, respectively, offering a comprehensive basis for assessing sequence effectiveness. You can access the data and code by visiting the URL `https://github.com/brandontrabucco/design-bench`.

B EXPERIMENTAL DETAILS

In this section, we provide further details about the experiments, including the URL for the code of the comparison algorithms, and the computational resources used.

B.1 METHODS

This section introduces approaches for offline optimization. To establish a baseline for future comparisons, we benchmark several recent offline MBO algorithms across our tasks. Existing methods fall into three main categories: forward, generative, and evolutionary methods. Forward methods focus on training robust surrogate models to combat adversarial optimization of inputs, followed by gradient-based maximization. Generative methods sample solutions from learned generative models with regularization. Evolutionary methods use neural networks and other techniques to learn proxy models, which are then solved using evolutionary algorithms. In addition, we also introduce traditional black box optimization methods for comparison. For evaluation, we consider three key components of offline optimization: model architecture, learning algorithm, and search algorithm, as outlined below.

B.1.1 TRADITIONAL BLACK BOX OPTIMIZATION METHODS

Algorithm 1 Offline Bayesian Optimization Trabucco et al. (2022)

1: Train a surrogate model $\hat{f}$ based on offline dataset $\mathcal{D}$.
2: Select the top 1 initial designs $\hat{\mathcal{D}} = (\mathbf{X}_t, \mathbf{y}_t)$ from the offline dataset $\mathcal{D}$.
3: **for** $t = 1...K$ **do**
4: Find $\mathbf{x}_t$ by optimizing the quasi-Expected-Improvemnt acquation function over the Gaussian Process: $\mathbf{x}_t = \arg\max_{\mathbf{x}} u(\mathbf{x} \mid \hat{\mathcal{D}}_{1:t-1})$.
5: Sample the Surrogate function: $y_t = \hat{f}(\mathbf{x}_t)$.
6: Augment the data $\hat{\mathcal{D}}_{1:t} = \{\hat{\mathcal{D}}_{1:t-1}, (\mathbf{x}_t, y_t)\}$ and update the Gaussian Process.
7: **end for**

BO-qEI Balandat et al. (2019): We perform offline Bayesian optimization by fitting a Gaussian Process using a learned surrogate model $\hat{f}(\mathbf{x})$. Next, we employ the quasi-Expected Improvement acquisition function to propose candidate solutions for efficiency. After the Bayesian optimization cycle is completed, we select the best candidates $\mathbf{x}$ from the dataset $\hat{\mathcal{D}}$. The procedure of BO-qEI is shown in Algorithm 1. We use the implementation from `https://github.com/brandontrabucco/design-baselines/tree/master/design_baselines/bo_qei`.

CMA-ES Hansen (2006): We perform offline optimization using the Covariance Matrix Adaptation Evolution Strategy (CMA-ES), which adapts the covariance matrix of a multivariate normal distribution to guide the search. At each iteration, candidate solutions are sampled from the distribution, and the distribution parameters are updated based on selected high-performing solutions. After completing the optimization cycle, we select the best candidates $\mathbf{x}$ from the dataset $\mathcal{D}$. The procedure of CMA-ES is shown in Algorithm 1. We use the implementation from `https://github.com/brandontrabucco/design-baselines/tree/master/design_baselines/cbas`.

B.1.2 FORWARD METHODS

ARCOO Lu et al. (2023): ARCOO learns both a surrogate model and an energy-based model, which characterizes the risk of degradation to address the out-of-distribution issue. The optimizer at each step is regulated by a risk suppression factor derived from the energy-based model. The procedure is outlined in Algorithm 3. The implementation can be found at `https://github.com/luhuakang/ARCOO`.

Algorithm 2 Covariance Matrix Adaptation Evolution Strategy (CMA-ES) Hansen (2006)

Initialize mean $\mathbf{m}_0$, step size σ_0, population size λ, initial covariance matrix $\mathbf{C}_0 = \mathbf{I}$
Set learning rates for mean (μ), covariance matrix (α), and step size adaptation (β)
for each generation $t = 0, 1, 2, \ldots$ **do**
  Sample λ candidate solutions $\mathbf{x}_i^{(t)} \sim \mathcal{N}(\mathbf{m}_t, \sigma_t^2 \mathbf{C}_t)$ for $i = 1, \ldots, \lambda$
  Evaluate fitness $f(\mathbf{x}_i^{(t)})$ for each candidate $\mathbf{x}_i^{(t)}$
  Select the top μ candidates with the best fitness values
  Update mean $\mathbf{m}_{t+1}$ as the weighted average of the top μ candidates:

$$\mathbf{m}_{t+1} = \sum_{i=1}^{\mu} w_i \mathbf{x}_i^{(t)}$$

  Update covariance matrix $\mathbf{C}_{t+1}$ to adapt the search distribution:

$$\mathbf{C}_{t+1} = (1 - \alpha)\mathbf{C}_t + \alpha \sum_{i=1}^{\mu} w_i \left(\mathbf{x}_i^{(t)} - \mathbf{m}_t\right) \left(\mathbf{x}_i^{(t)} - \mathbf{m}_t\right)^T$$

  Update step size σ_{t+1} based on the success rate of candidate solutions:

$$\sigma_{t+1} = \sigma_t \exp\left(\beta \left(\frac{\|\mathbf{p}_\sigma\|}{\|\mathcal{N}(0, \mathbf{I})\|} - 1\right)\right)$$

  Update evolution path $\mathbf{p}_\sigma$ for step size adaptation
end for
return Best solutions found during the optimization process

Tri-mentoring Chen et al. (2023): Tri-mentoring trains three surrogate models on the offline dataset and utilizes majority voting to generate consensus labels. To mitigate potential errors in the consensus, an adaptive soft-labeling module is applied. The complete algorithm is presented in Algorithm 4. The implementation can be found at https://github.com/GGchen1997/parallel_mentoring.

B.1.3 GENERATIVE METHODS

Autofocusing CbAS Brookes et al. (2019): The Autofocusing CbAS iteratively updates a search model for optimal design by focusing on high-quality solutions. It employs importance sampling to weight samples based on their likelihood of meeting desired conditions, allowing for targeted optimization. Using weighted Maximum Likelihood Estimation (MLE), the search model is refined, while an oracle model is retrained to adaptively reduce bias as the design space is explored. The procedure of Autofocusing CbAS is shown in Algorithm 5. We use the implementation from https://github.com/brandontrabucco/design-baselines/tree/master/design_baselines/autofocused_cbas.

B.1.4 EVOLUTIONARY METHODS

TTDDEA Huang et al. (2021): TTDDEA is an offline optimization method that combines tri-training with evolutionary algorithms. It uses three Radial Basis Function Networks (RBFNs) as surrogate models to predict fitness scores and generates high-confidence pseudo-labels to augment limited training data. By applying an evolutionary optimization process, TT-DDEA iteratively updates these surrogate models and selects high-performing candidate solutions for the next generation. This approach leverages semi-supervised learning and multi-model ensembles to address data scarcity in offline environments. The algorithm can be seen in Algorithm 6. We use the implementation from https://github.com/HandingWangXDGroup/TT-DDEA.

Algorithm 3 Accumulative Risk Controlled Offline Optimization (ARCOO) Lu et al. (2023)

Input:Offline dataset $\mathcal{D}$, learning rate η, maximum Langevin dynamics step K, Langevin dynamics stepsize λ, and initial momentum m.

1: Initialize dual-head model that consists of surrogate head $\hat{f}_\theta(\mathbf{x})$ and energy head $E_\phi(\mathbf{x})$.
2: **for** each training epoch **do**
3: Update $\hat{f}_\theta(\mathbf{x})$ using MSE loss:

$$\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}_\mathcal{D}(\theta)$$

4: Sample high-risk distribution $q(\mathbf{x})$ by Langevin dynamics: $q(\mathbf{x}) = \text{LD}_\theta(p(\mathbf{x}); K)$, i.e.,

$$\mathbf{x}_k \leftarrow \mathbf{x}_{k-1} + \lambda \nabla_\mathbf{x} \hat{f}_\theta(\mathbf{x}_{k-1}) + \omega_k, \quad k = 1, \ldots, K,$$

where $\omega_k \sim \mathcal{N}(0, \lambda)$, and $\mathbf{x}_0 \sim p(\mathbf{x})$. Sampling starts from the low-risk empirical distribution $p(\mathbf{x})$ over the offline dataset.
5: Update $E_\phi(\mathbf{x})$ using contrastive divergence loss:

$$\phi \leftarrow \phi - \eta \nabla_\phi \left[\text{KL}(p(\mathbf{x}) \| h_\phi(\mathbf{x})) - \text{KL}(q(\mathbf{x}) \| h_\phi(\mathbf{x})) \right],$$

where h_ϕ is derived from $E_\phi(\mathbf{x})$.
6: **end for**
7: Let $\tilde{\mathcal{P}}$ be an empirical distribution over a batch of the high-quality solutions in $\mathcal{D}$, and $\tilde{\mathcal{Q}} = \text{LD}_\theta(\tilde{\mathcal{P}}; K)$.
8: Calculate the risk suppression factor:

$$R_\phi(\mathbf{x}) = m(E_{\tilde{\mathcal{Q}}} - E_\phi(\mathbf{x}))(E_{\tilde{\mathcal{Q}}} - E_{\tilde{\mathcal{P}}})^{-1}.$$

9: **for** $t = 1$ to T **do**
10: $\mathbf{x}_t \leftarrow \mathbf{x}_{t-1} + R_\phi(\mathbf{x}_{t-1}) \nabla_\mathbf{x} \hat{f}_\theta(\mathbf{x}_{t-1})$.
11: **end for**
12: **return** Final solution $\mathbf{x}_{\text{app}} = \mathbf{x}_T$ for online application.

Algorithm 4 Tri-mentoring for Offline Model-based Optimization Chen et al. (2023)

Input: The static dataset $\mathcal{D}$, the number of iterations T, the optimizer OPT$(\cdot)$.

1: Initialize x_0 as the design with the highest score in $\mathcal{D}$.
2: Train proxies $f_\theta^A(\cdot)$, $f_\theta^B(\cdot)$, and $f_\theta^C(\cdot)$ on $\mathcal{D}$ with different initializations.
3: **for** $t \leftarrow 0$ to $T-1$ **do**
4: Sample K neighborhood points at x_t as $S(\mathbf{x}_t)$.
5: Compute pairwise comparison labels $\hat{y}^A$, $\hat{y}^B$, and $\hat{y}^C$ for the three proxies on $S(\mathbf{x}_t)$.
6: Derive consensus labels: $\hat{y}^V = \text{majority\_voting}(\hat{y}^A, \hat{y}^B, \hat{y}^C)$.
7: **for** proxy in $\{f_\theta^A(\cdot), f_\theta^B(\cdot), f_\theta^C(\cdot)\}$ **do**
8: Initialize soft-labels as consensus labels: $\hat{y}^S = \hat{y}^V$.
9: *Inner level*: fine-tune the proxy with Eq. (8).
10: *Outer level*: learn more accurate soft-labels $\hat{y}^S$ with Eq. (9).
11: Mentor proxy using the optimized soft-labels $\hat{y}^S$ with Eq. (8).
12: **end for**
13: Form a more robust ensemble as $f_\theta(x) = \frac{1}{3}\left(f_\theta^A(\mathbf{x}) + f_\theta^B(\mathbf{x}) + f_\theta^C(\mathbf{x})\right)$.
14: Gradient ascent: $\mathbf{x}_{t+1} = \mathbf{x}_t + \eta \text{OPT}(\nabla_\mathbf{x} f_\theta(\mathbf{x}_t))$.
15: **end for**
16: **return** The high-scoring designs $\mathbf{x}^* = \mathbf{x}_T$.

CCDDEA Gong et al. (2023): CCDDEA combines hierarchical surrogate models with cooperative coevolution to solve large-scale optimization problems. It uses a global model (HM) and local models (LMs) to guide search at different levels. Local searches are enhanced with gradient-based and evolutionary operators, while global communication merges sub-populations. The dynamic space division strategy improves convergence by shifting focus from local to global optimization.

Algorithm 5 Autofocused Model-based Optimization (Autofocusing CbAS) Brookes et al. (2019)

Input: Offline dataset, $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$; oracle model class, $p_\beta(y \mid x)$ with parameters, β, that can be estimated with MLE; search model class, $p_\theta(x)$ with parameters, θ, that can be estimated with weighted MLE or approximations thereof; desired constraint set, S (e.g., $S = \{y \mid y \geq y_\tau\}$); maximum number of iterations, T; number of samples to generate, m; EDA-specific monotonic transformation, $V(\cdot)$.

Initialization: Obtain $p_\theta(x)$ by fitting to $\{x_i\}_{i=1}^{n}$ with the search model class. For the search model, set $p_{\theta(0)}(x) \leftarrow p_\theta(x)$. For the oracle, $p_{\beta(0)}(y \mid x)$, use MLE with equally weighted training data.

1: **for** $t = 1, \ldots, T$ **do**

2: Sample from the current search model, $\{\tilde{x}_i^{(t)}\}_{i=1}^{m} \sim p_{\theta(t-1)}(x),\ \forall i \in \{1, \ldots, m\}$.

3: $v_i \leftarrow V\left(P_{\beta(t-1)}(y \in S \mid \tilde{x}_i^{(t)})\right),\ \forall i \in \{1, \ldots, m\}$.

4: Fit the updated search model, $p_{\theta(t)}(x)$, using weighted MLE with the samples, $\{\tilde{x}_i^{(t)}\}_{i=1}^{m}$, and their corresponding EDA weights, $\{v_i\}_{i=1}^{m}$.

5: Compute importance weights for the training data, $w_i \leftarrow p_{\theta(t)}(x_i)/p_{\theta(0)}(x_i),\ i = 1, \ldots, n$.

6: Retrain the oracle using the re-weighted training data,

$$\beta(t) \leftarrow \arg\max_{\beta \in B} \frac{1}{n} \sum_{i=1}^{n} w_i \log p_\beta(y_i \mid x_i).$$

7: **end for**

8: **return** The most promising candidates among $\{\tilde{x}_i^{(t)}, ..., \tilde{x}_m^{(t)}\}_{t=1}^{T}$.

Algorithm 6 Tri-Training Data-Driven Evolutionary Algorithm (TT-DDEA) Huang et al. (2021)

Input: Separate offline data sets L_1, L_2, and L_3, trained RBFNs M_1, M_2, and M_3, current population P, population size Q.

1: **for** $i = 1 \rightarrow 3$ **do**

2: Use the models M_1, M_2, M_3 to predict the fitness of population, written as $f_1^1, f_2^1, \ldots, f_Q^1$, $f_1^2, f_2^2, \ldots, f_Q^2$, and $f_1^3, f_2^3, \ldots, f_Q^3$.

3: **end for**

4: **for** $i = 1 \rightarrow 3$ **do**

5: Find the high-confidence data x_i as in Equation (9).

6: Calculate its pseudo label $\hat{y}_i$ as in Equation (10).

7: Update L_i to L_i' by $(x_i, \hat{y}_i)$ as in Equation (11).

8: **end for**

9: Train RBFNs M_1, M_2, and M_3 with L_1', L_2', and L_3'.

10: Use the evolutionary algorithms to obtain promising candidates.

11: **return** The most promising candidates found by the evolutionary algorithms.

The procedure of CCDDEA is shown in Algorithm. The implementation can be found in https://github.com/LabGong/cc-ddea.

Algorithm 7 Offline Data-Driven Optimization at Scale: A Cooperative Coevolutionary Approach (CC-DDEA) Gong et al. (2023)

Input: $\mathcal{D}$: The offline data;
n: The size of complete population;
T_H, T_L: The maximum generations of higher-level and lower-level optimization;
g_{init}: The initial number of groups;
T_g: The interval for updating the number of groups;
T_r: The interval for re-dividing the sub-spaces;
α: The learning rate of gradient descent;
r_{top}: The control parameter for top-ranked random merging in the cooperative search;
Output: The best solution
Initialization: $i_r \leftarrow 0$ (index of T_r), $g \leftarrow g_{init}$
$P \leftarrow$ Latin hypercube sampling
for $i \leftarrow 1$ to T_H **do**
  if $i > 1$ and $i \bmod T_g = 1$ and $g > 1$ **then**
    $g \leftarrow g - 1$
    $i_r \leftarrow 0$
  end if
  if $i_r \bmod T_r = 0$ and $g > 1$ **then**
    $G \leftarrow$ division rules
    $(HM, LMi) \leftarrow$ HSJL$(\mathcal{D}, G)$
  end if
  $SP \leftarrow$ split P according to G
  for $SP_j \in SP$ **do**
    $SP_j \leftarrow$ LowerLevelSearch$(SP_j, T_L, \alpha, LM_j)$
  end for
  if $g > 1$ **then**
    $P \leftarrow$ HigherLevelSearch(SP, r_{top}, HM, LMs)
  end if
  $i_r \leftarrow i_r + 1$
end for
return $P[0]$

B.1.5 Constrained Methods

We selected two primary categories of offline constraint algorithms: deep learning-based constrained offline methods and evolution-based constrained offline methods. For our experiments, we utilized classic methods from both categories.

CARCOO: This is a simplified constrained version of ARCOO. We integrate the degree of constraint violation into risk assessment, treating solutions that violate constraints as high risk. CARCOO employs the three models (i.e., a surrogate model, an energy-based model to characterize OOD risk, an energy-based model to characterize constrained risk) to train the models on a training dataset. The algorithm can be found in Algorithm 8.

CCOMs: This is a simple experimental version of PRIME (Kumar et al., 2022). We use $\theta^* = \arg\min_\theta \mathbb{E}_{\mathbf{x}_i, y_i \sim \mathcal{D}_{\text{feasible}}} \left[\left(f_\theta \left(\mathbf{x}_i \right) - y_i \right)^2 \right] - \alpha \mathbb{E}_{\mathbf{x}_i^- \sim \mathcal{D}_{\text{infeasible}}} \left[f_\theta \left(\mathbf{x}_i^- \right) \right]$. This means that in addition to fitting the surrogate model, we also make the values of points that violate the constraints as small as possible. Then use the conservative model COMs (Trabucco et al., 2021) as surrogate model to train. Pseudocode can be found at Algorithm 9.

DDEA-PF & DDEA-SPF Huang & Wang (2021b): DDEA-PF and DDEA-SPF are two data-driven evolutionary algorithms for handling constraints in optimization tasks. DDEA-PF employs penalty

Algorithm 8 Constrained Accumulative Risk Controlled Offline Optimization (CARCOO)

Input:Offline dataset $\mathcal{D}$, learning rate η, maximum Langevin dynamics step K, Langevin dynamics stepsize λ, and initial momentum m.

1: Initialize dual-head model that consists of surrogate head $\hat{f}_\theta(\mathbf{x})$ and energy head $E_\phi(\mathbf{x})$.
2: **for** each training epoch **do**
3: Update $\hat{f}_\theta(\mathbf{x})$ using MSE loss:

$$\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}_\mathcal{D}(\theta)$$

4: Sample high-risk distribution $q(\mathbf{x})$ by Langevin dynamics: $q(\mathbf{x}) = \text{LD}_\theta(p(\mathbf{x}); K)$, i.e.,

$$\mathbf{x}_k \leftarrow \mathbf{x}_{k-1} + \lambda \nabla_\mathbf{x} \hat{f}_\theta(\mathbf{x}_{k-1}) + \omega_k, \quad k = 1, \ldots, K,$$

where $\omega_k \sim \mathcal{N}(0, \lambda)$, and $\mathbf{x}_0 \sim p(\mathbf{x})$. Sampling starts from the low-risk empirical distribution $p(x)$ over the offline dataset.
5: Update $E_\phi(\mathbf{x})$ using contrastive divergence loss:

$$\phi \leftarrow \phi - \eta \nabla_\phi \left[\text{KL}(p(\mathbf{x}) \| h_\phi(\mathbf{x})) - \text{KL}(q(\mathbf{x}) \| h_\phi(\mathbf{x})) \right],$$

where h_ϕ is derived from $E_\phi(\mathbf{x})$.
6: Update $\hat{E}_\tau(\mathbf{x})$ using contrastive divergence loss:

$$\tau \leftarrow \tau - \eta \nabla_\tau \left[\text{KL}(\hat{p}(\mathbf{x}) \| \hat{h}_\tau(\mathbf{x})) - \text{KL}(\hat{q}(\mathbf{x}) \| \hat{h}_\tau(\mathbf{x})) \right],$$

where $\hat{h}_\tau$ is derived from $\hat{E}_\phi(\mathbf{x})$, $\hat{p}(\mathbf{x})$ is solutions that satisfy the constraints in the dataset, $\hat{q}(\mathbf{x})$ is solutions in the dataset that do not satisfy the constraints
7: **end for**
8: Let $\tilde{\mathcal{P}}$ be an empirical distribution over a batch of the high-quality solutions in $\mathcal{D}$, and $\tilde{\mathcal{Q}} = \text{LD}_\theta(\tilde{\mathcal{P}}; K)$.
9: Calculate the risk suppression factor:

$$R_\phi(\mathbf{x}) = m(E_{\tilde{\mathcal{Q}}} - E_\phi(\mathbf{x}))(E_{\tilde{\mathcal{Q}}} - E_{\tilde{\mathcal{P}}})^{-1}.$$

$$R_\tau(\mathbf{x}) = m(E_{\tilde{\mathcal{Q}}} - E_\tau(\mathbf{x}))(E_{\tilde{\mathcal{Q}}} - E_{\tilde{\mathcal{P}}})^{-1}.$$

10: **for** $t = 1$ to T **do**
11: $\mathbf{x}_t \leftarrow \mathbf{x}_{t-1} + \frac{(R_\phi(\mathbf{x}_{t-1}) + R_\tau(\mathbf{x}_{t-1}))}{2} \nabla_\mathbf{x} \hat{f}_\theta(\mathbf{x}_{t-1})$.
12: **end for**
13: **return** Final solution $\mathbf{x}_{\text{app}} = \mathbf{x}_T$ for online application.

functions to manage constraints, adjusting the penalty to push solutions towards feasibility. On the other hand, DDEA-SPF focuses on adaptive penalty, i.e., $\text{fitness}(x) = f(\mathbf{x}) + r_i \frac{(\sum_{j=1}^{M} \max[0, g_j(\mathbf{x})]^q)}{g_{\max,j}}$. DDEA-PF is presented in Algorithm 10. The implementation is from `https://github.com/HandingWangXDGroup/Constraint-Handling-OfflineDDEA`.

B.2 Computation Resources

The computing resources required for the research described in this paper are relatively modest, requiring only a single Nvidia GeForce RTX 3090 GPU. The experiments were efficiently completed using this powerful graphics card, almost all within a 24-hour timeframe.

C Experimental Results

Below, we present the experimental results of our study, providing detailed insights and analyses in the following sections. The experiments were designed to evaluate the performance of our proposed methods under various conditions. All results are averaged over eight times. It is important to note that under different seeds, some may be able to run relatively good experimental results, while others

Algorithm 9 Constrained Conservative Objective Models for Offline Optimization (CCOMs)

Input: Offline dataset $\mathcal{D}$, the learning rate of OOD gradient ascent η, trade-off coefficient α.
Initialization: Surrogate model $\hat{f}_\theta$.

1: **for** $i = 1$ to training_steps **do**
2: Sample $(\mathbf{x}_0, y) \sim \mathcal{D}$
3: Find $\mathbf{x}_T(x_0)$ via gradient ascent from $\mathbf{x}_0$:

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \eta \nabla_{\mathbf{x}} \hat{f}_\theta(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_t}; \quad \mu(\mathbf{x}) = \sum_{\mathbf{x}_0 \in \mathcal{D}} \delta_{\mathbf{x}=\mathbf{x}_T(\mathbf{x}_0)}.$$

4: Minimize $\mathcal{L}(\theta; \alpha)$ with respect to θ.

$$\mathcal{L}(\theta;\alpha) = \mathbb{E}_{\mathbf{x}_0 \sim \mathcal{D}}\left(\hat{f}_\theta(\mathbf{x}_0) - y\right)^2 - \alpha \mathbb{E}_{\mathbf{x}_0}\left[\hat{f}_\theta(\mathbf{x}_0)\right] + \alpha \mathbb{E}_{\mu(\mathbf{x})}\left[\hat{f}_\theta(\mathbf{x})\right] - \mathbb{E}_{\mathbf{x}^- \sim \mathcal{D}_{\text{inf}}}\left[f_\theta\left(\mathbf{x}^-\right)\right]$$

$$\theta \leftarrow \theta - \lambda \nabla_\theta \mathcal{L}(\theta; \alpha)$$

5: **end for**
6: Initialize optimizer at the optimum in $\mathcal{D}$:
7: Find $\mathbf{x}^\star$ via trust-region gradient ascent from $\tilde{\mathbf{x}}$:

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \eta \nabla_x \mathcal{L}_{\text{opt}}(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_t}$$

where $\mathcal{L}_{\text{opt}}(\mathbf{x}) := \hat{f}^\star_\theta(\mathbf{x})$.
8: Return the solution $\mathbf{x}^\star = \mathbf{x}_T$.

Algorithm 10 Data-Driven Evolutionary Optimization with Penalty Function (DDEA-PF) Huang & Wang (2021b)

1: Initialize population P with size Q
2: Set the degree of violation of the jth constraint $g_j(\mathbf{x})$, penalty coefficient r_i.
3: **for** each generation T **do**
4: **for** each individual x in P **do**
5: Evaluate fitness using the objective function with a penalty for constraint violation:

$$\text{fitness}(x) = f(\mathbf{x}) + r_i(\sum_{j=1}^{M} \max[0, g_j(\mathbf{x})]^q)$$

6: **end for**
7: Perform selection, crossover, and mutation in population P to find best feasible solutions.
8: **end for**
9: **return** Best feasible solutions $\mathbf{x}^* = \mathbf{x}_T$.

may not be able to surpass the offline dataset. When such cases are relatively rare (i.e., less than 4 times), we will delete these cases, otherwise we will consider SI to be $-\infty$ at this time.

The experimental results in the unconstrained scenario show that TTDDEA and CBAS algorithms are the most balanced and stable on various benchmark functions, demonstrating high performance and stability. Although CMA-ES performs well in some cases, the stability of the results is poor. ARCOO and BO perform well on most functions.

The experimental results in the constraint scenario show that the DE-PF and DE-SFP algorithms do not perform well on various benchmark functions. In addition, the performance of DE-PF and DE-SFP on multiple benchmark functions is almost the same, showing consistent performance. The performance of the CCOMS algorithm fluctuates greatly in different dimensions and functions. Overall, the CARCOO algorithm has good performance and stability when dealing with constrained optimization problems, providing an important reference for subsequent algorithm optimization.

In the hybrid vehicle task, we found that some tasks could not be completed within 24 hours, highlighting the limitations of the current offline optimization algorithms. These algorithms struggle with high-dimensional and time-consuming tasks, emphasizing the need to reduce the number and points of evaluations. Streamlining these aspects is crucial for improving efficiency and effectiveness in such complex optimization scenarios. Most algorithms that can be completed within the specified time can achieve significant improvements. For example, when the 0-60 loss is severe, BO improves from 257 to 33, with an SI of 0.91. ARCOO improves to 17, with an SI of 0.92. However, other algorithms either cannot produce results within the specified time or fail to improve, resulting in a negative SI. We found that many algorithms cannot find feasible solutions or solutions that are better than offline datasets for complex tasks. Therefore, how to find the optimal solution with stable improvement in future work is an issue worth considering.

Table 5: Overall results in GTOPX unconstriant scenario. Details are the same as main body. The symbol "$-\infty$" means that the algorithm can't find solutions exceeding $f(\boldsymbol{x}^*_{\mathrm{OFF}})$. For each task, algorithms within one standard deviation of having the highest performance are **bolded**.

	Tasks	GTOPX 2		GTOPX 3		GTOPX 4		GTOPX 6	
	Metrics	FS ↓	SI ↑	FS ↓	SI ↑	FS ↓	SI ↑	FS ↓	SI ↑
10-60	$f(\boldsymbol{x}^*_{\mathrm{OFF}})$	276.47		228.02		322.74		142.05	
	CMA-ES	276.47±1.76	0.00±0.00	228.02±2.27	0.00±0.00	322.74±1.11	0.00±0.00	142.05±0.48	0.00±0.00
	Tri-mentoring	251.66±30.75	0.46±0.45	145.43±70.45	0.68±0.33	247.04±93.80	0.53±0.38	128.37±25.11	0.55±0.41
	CBAS	276.47±1.76	$-\infty$	117.26±56.17	**0.82±0.05**	202.76±64.70	0.21±0.05	194.46±45.24	$-\infty$
	ARCOO	**82.27±9.74**	**0.89±0.02**	116.75±59.80	0.79±0.07	198.45±102.26	0.76±0.12	**64.30±19.63**	**0.82±0.03**
	TTDDEA	472.87±530.28	0.74±2.24	829.88±1183.97	-0.46±3.44	1634.61±1578.12	$-\infty$	145.75±43.06	-0.07±0.64
	BO	**82.98±18.24**	**0.81±0.02**	**69.69±15.41**	**0.82±0.01**	**105.36±23.90**	**0.81±0.03**	**57.95±13.14**	0.73±0.02
	CCDDEA	276.57±4.77	$-\infty$	228.19±8.06	$-\infty$	326.68±5.93	$-\infty$	155.44±33.21	$-\infty$
20-70	$f(\boldsymbol{x}^*_{\mathrm{OFF}})$	219.12		171.28		242.26		121.48	
	CMA-ES	219.12±1.42	0.00±0.00	171.28±0.57	0.00±0.00	242.26±0.71	0.00±0.00	121.48±0.50	0.00±0.00
	Tri-mentoring	166.90±63.81	0.74±0.30	154.24±20.72	0.51±0.41	174.32±56.98	**0.75±0.31**	104.82±18.42	0.42±0.38
	CBAS	219.12±1.42	0.00±0.00	92.08±14.34	0.75±0.04	242.26±0.71	0.13±0.02	90.02±38.85	0.32±0.10
	ARCOO	**80.06±6.65**	**0.83±0.01**	**58.01±7.03**	**0.87±0.03**	138.43±32.21	**0.75±0.07**	**62.79±19.66**	**0.73±0.07**
	TTDDEA	248.55±43.15	-0.68±0.12	1234.48±1688.45	$-\infty$	989.33±1476.65	$-\infty$	117.25±37.72	-0.94±1.91
	BO	**84.13±4.97**	0.74±0.30	**59.46±10.87**	0.76±0.03	**107.50±18.28**	0.72±0.03	71.90±13.96	0.69±0.01
	CCDDEA	219.22±3.95	$-\infty$	172.36±5.27	$-\infty$	242.95±4.04	$-\infty$	121.70±1.72	$-\infty$
30-80	$f(\boldsymbol{x}^*_{\mathrm{OFF}})$	175.46		134.75		194.36		102.98	
	CMA-ES	175.46±0.69	0.00±0.00	134.75±0.64	0.00±0.00	194.36±0.37	0.00±0.00	102.98±0.36	0.00±0.00
	Tri-mentoring	138.32±49.95	0.40±0.45	125.84±17.10	0.55±0.40	158.57±20.94	0.52±0.40	102.98±0.36	0.23±0.38
	CBAS	175.46±0.69	0.00±0.00	96.61±20.89	0.67±0.02	190.91±7.41	0.12±0.02	90.94±24.05	0.40±0.05
	ARCOO	**86.91±16.70**	**0.78±0.05**	**65.17±14.18**	**0.80±0.05**	**110.97±8.00**	**0.74±0.05**	**58.05±10.68**	**0.72±0.11**
	TTDDEA	194.83±30.25	-8.67±2.67	198.92±96.66	$-\infty$	458.64±323.03	$-\infty$	167.23±62.22	$-\infty$
	BO	108.25±25.51	0.66±0.03	72.75±14.79	0.65±0.02	127.51±15.86	0.63±0.03	60.48±6.68	0.57±0.03
	CCDDEA	176.53±3.76	$-\infty$	135.21±2.74	$-\infty$	193.12±2.67	$-\infty$	116.20±33.92	$-\infty$
40-90	$f(\boldsymbol{x}^*_{\mathrm{OFF}})$	136.50		102.75		153.91		83.44	
	CMA-ES	136.50±0.28	0.00±0.00	102.75±0.53	0.00±0.00	153.91±0.29	0.00±0.00	83.44±0.21	0.00±0.00
	Tri-mentoring	127.39±18.01	0.51±0.44	96.53±12.48	0.51±0.43	153.91±0.41	0.43±0.47	83.44±0.21	0.41±0.48
	CBAS	136.50±0.28	0.00±0.00	90.01±10.47	0.43±0.04	153.91±0.41	0.07±0.02	69.75±17.59	0.34±0.09
	ARCOO	**77.72±15.90**	**0.78±0.06**	**59.82±8.20**	**0.73±0.06**	104.88±21.71	**0.66±0.03**	63.55±10.81	**0.60±0.20**
	TTDDEA	280.25±223.36	-37.89±21.67	1961.71±3763.00	$-\infty$	299.17±120.87	$-\infty$	831.60±1441.82	$-\infty$
	BO	84.37±14.90	0.48±0.02	85.13±16.80	0.52±0.02	**90.88±10.68**	0.46±0.06	**53.83±9.09**	0.47±0.03
	CCDDEA	138.24±2.59	$-\infty$	102.86±2.54	$-\infty$	153.43±3.53	$-\infty$	83.20±1.05	$-\infty$

D ADDITIONAL EXPERIMENTS

Since we have the flexibility to construct the distribution of dataset, we also offer two additional common settings for complex data environments. The first setting involves adjusting the size of the

Table 6: Overall results in GTOPX constrianed scenario. Details are the same as Table 5. The symbol "-" means that the algorithm cannot work because of too few solutions that satisfly the constraints.

	Tasks	GTOPX 1		GTOPX 5		GTOPX 7	
	Metrics	FS ↓	SI ↑	FS ↓	SI ↑	FS ↓	SI ↑
10-60	$f(\boldsymbol{x}^*_{\text{OFF}})$	114.86		-		568.18	
	DE-PF	530.92±1.50	$-\infty$	-	-	1366.08±3.42	$-\infty$
	DE-SPF	530.92±1.50	$-\infty$	-	-	1366.08±3.42	$-\infty$
	CARCOO	111.07±4.18	**0.27±0.21**	-	-	**64.07±31.42**	0.91±0.03
	CCOMs	**31.65±0.18**	-0.70±1.13	-	-	545.49±8.81	**1.00±0.00**
20-70	$f(\boldsymbol{x}^*_{\text{OFF}})$	87.15		-		422.76	
	DE-PF	392.04±4.10	$-\infty$	-	-	1044.00±5.69	$-\infty$
	DE-SPF	392.04±4.10	$-\infty$	-	-	1044.00±5.69	$-\infty$
	CARCOO	**82.04±5.33**	**0.35±0.19**	-	-	199.37±136.86	0.79±0.07
	CCOMs	248.57±177.55	-0.75±0.92	-	-	**30.50±0.49**	**1.00±0.00**
25-75	$f(\boldsymbol{x}^*_{\text{OFF}})$	75.13		-		346.52	
	DE-PF	283.39±14.47	$-\infty$	0	$-\infty$	954.35±3.16	$-\infty$
	DE-SPF	263.97±52.96	$-\infty$	0	$-\infty$	954.35±3.16	$-\infty$
	CARCOO	74.98±1.09	**-3.71±9.62**	**-58742.77±23361.35**	**0.37±0.12**	260.16±109.21	0.62±0.24
	CCOMs	164.89±129.51	0.62±0.24	-12634.47±22004.53	-0.19±0.34	**954.35±3.16**	$-\infty$
30-80	$f(\boldsymbol{x}^*_{\text{OFF}})$	65.19		-		276.33	
	DE-PF	234.79±1.10	$-\infty$	-	-	890.64±3.36	$-\infty$
	DE-SPF	234.79±1.10	$-\infty$	-	-	890.64±3.36	$-\infty$
	CARCOO	**62.46±2.41**	**0.37±0.16**	**19.78±0.00**	**0.13±0.00**	150.00±65.72	0.66±0.14
	CCOMs	112.97±99.94	-1.93±0.30	-	-	**30.50±0.49**	**1.00±0.00**
40-90	$f(\boldsymbol{x}^*_{\text{OFF}})$	46.84		-		148.7	
	DE-PF	203.64±0.75	$-\infty$	-	-	780.34±3.49	$-\infty$
	DE-SPF	203.64±0.75	$-\infty$	-	-	780.34±3.49	$-\infty$
	CARCOO	**46.56±2.68**	**0.17±0.22**	**1051.48±14.50**	$-\infty$	143.31±8.99	0.22±0.13
	CCOMs	203.64±0.75	-9.77±11.00	-	-	**114.93±16.23**	**1.00±0.00**

Table 7: Overall results in CEC constrianed scenario. Details are the same as Table 6.

	Tasks	CEC 1		CEC 2		CEC 3		CEC 4		CEC 5	
	Metrics	FS ↓	SI ↑	FS ↓	SI ↑	FS ↓	SI ↑	FS ↓	SI ↑	FS ↓	SI ↑
10-60	$f(\boldsymbol{x}^*_{\text{OFF}})$	-		1.08		2.12		264.42		8.61	
	DE-PF	-	-	**1.34±0.01**	0.00±0.00	3.38±0.26	$-\infty$	381.04±3.52	$-\infty$	30.97±3.23	$-\infty$
	DE-SPF	-	-	**1.34±0.01**	0.00±0.00	3.37±0.01	$-\infty$	**277.45±19.22**	$-\infty$	29.44±0.00	0.00±0.00
	CARCOO	-	-	1.17±0.25	-0.59±2.04	**2.65±0.01**	**0.70±0.00**	-	-	**11.33±0.41**	**0.87±0.12**
	CCOMs	-	-	422.29±159.58	$-\infty$	2.72±0.15	0.06±0.01	-	-	-	-
20-70	$f(\boldsymbol{x}^*_{\text{OFF}})$	-		1.08		2.00		264.42		6.75	
	DE-PF	-	-	**1.32±0.02**	$-\infty$	3.03±0.06	$-\infty$	378.29±1.61	$-\infty$	27.86±0.19	$-\infty$
	DE-SPF	-	-	**1.32±0.02**	$-\infty$	3.01±0.01	$-\infty$	**274.70±12.15**	$-\infty$	18.31±5.31	$-\infty$
	CARCOO	-	-	2.32±0.06	$-\infty$	**0.67±0.29**	0.86±0.07	-	-	**14.75±0.03**	**0.83±0.01**
	CCOMs	-	-	46.77±32.15	$-\infty$	3.01±0.00	**-0.01±0.00**	302.93±11.67	$-\infty$	-	-
25-75	$f(\boldsymbol{x}^*_{\text{OFF}})$	-		1.08		2.00		264.42		5.84	
	DE-PF	-	-	**1.30±0.03**	$-\infty$	3.00±0.13	$-\infty$	377.28±0.24	$-\infty$	38.67±7.65	$-\infty$
	DE-SPF	-	-	**1.30±0.03**	$-\infty$	2.04±0.02	**0.54±0.01**	**270.70±3.74**	$-\infty$	17.99±0.11	$-\infty$
	CARCOO	-	-	2.43±0.07	$-\infty$	1.29±0.10	0.88±0.07	-	-	**10.37±0.90**	**-0.86±0.75**
	CCOMs	-	-	57.41±13.78	$-\infty$	**2.02±0.00**	0.08±0.00	297.66±3.34	0.43±0.39	-	-
30-80	$f(\boldsymbol{x}^*_{\text{OFF}})$	-		1.08		2.00		264.42		4.91	
	DE-PF	-	-	**1.27±0.03**	$-\infty$	2.90±0.18	$-\infty$	351.49±5.17	$-\infty$	20.41±0.01	$-\infty$
	DE-SPF	-	-	**1.27±0.03**	$-\infty$	2.84±0.14	$-\infty$	**270.70±3.74**	$-\infty$	16.08±0.23	$-\infty$
	CARCOO	-	-	2.79±0.22	$-\infty$	**2.00±0.00**	**0.94±0.00**	-	-	**9.18±0.07**	**-0.17±0.18**
	CCOMs	-	-	66.71±3.35	$-\infty$	2.14±0.01	0.08±0.01	286.00±0.00	$-\infty$	-	-
40-90	$f(\boldsymbol{x}^*_{\text{OFF}})$	-		1.08		2.00		264.42		2.95	
	DE-PF	-	-	**1.26±0.07**	$-\infty$	2.81±0.27	$-\infty$	303.41±35.33	$-\infty$	19.63±0.55	$-\infty$
	DE-SPF	-	-n	1.23±0.05	$-\infty$	2.01±0.12	0.32±0.15	**269.19±0.83**	$-\infty$	13.06±0.19	$-\infty$
	CARCOO	-	-	3.49±0.49	$-\infty$	**2.00±0.00**	**0.94±0.00**	-	-	**2.69±0.11**	**0.71±0.67**
	CCOMs	-	-	64.21±9.70	$-\infty$	2.20±0.00	-0.02±0.00	269.04±6.89	$-\infty$	-	-

Table 8: Overall results in PROTEIN unconstrianed & constrianed scenario.

	Tasks	Mujoco 1		Mujoco 2			Mujoco 1		Mujoco 2	
	$f(\boldsymbol{x}^*_{\text{OFF}})$	2.30		2.08			2.17		1.56	
	CMA-ES	-	-	-	-		-	-	-	-
	Tri-mentoring	**2.30±0.03**	0.00±0.00	-	-		-	-	-	-
10-60	CBAS	-	-	-	-		-	-	-	-
	ARCOO	**2.30±0.01**	0.04±0.01	**2.08±0.01**	0.02±0.01	0-90	**2.17±0.01**	0.01±0.01	**1.56±0.01**	0.01±0.01
	TTDDEA	-	-	-	-		-	-	-	-
	BO	-	-	-	-		-	-	-	-
	$f(\boldsymbol{x}^*_{\text{OFF}})$	2.00		1.88			1.83		1.33	
	CMA-ES	-	-	-	-		-	-	-	-
	Tri-mentoring	**2.00±0.02**	0.00±0.00	-	-		-	-	-	-
20-70	CBAS	-	-	-	-		-	-	-	-
	ARCOO	**2.00±0.01**	0.02±0.01	**1.88±0.01**	0.03±0.01	0-80	**1.83±0.01**	0.01±0.01	**1.33±0.01**	0.01±0.01
	TTDDEA	-	-	-	-		-	-	-	-
	BO	-	-	-	-		-	-	-	-
	$f(\boldsymbol{x}^*_{\text{OFF}})$	1.92		1.82						
	CMA-ES	-	-	-	-					
	Tri-mentoring	1.92±0.01	0.00±0.00	-	-					
25-75	CBAS	-	-	-	-					
	ARCOO	1.92±0.01	0.04±0.01	**1.82±0.01**	0.02±0.01					
	TTDDEA	-	-	-	-					
	BO	**8.51±0.00**	0.50±0.00	-	-					
	$f(\boldsymbol{x}^*_{\text{OFF}})$	1.86		1.70			1.66		1.17	
	CMA-ES	-	-	-	-		-	-	-	-
	Tri-mentoring	-	-	-	-		**1.66±0.00**	0.00±0.00	-	-
30-80	CBAS	-	-	-	-		-	-	-	-
	ARCOO	**1.86±0.01**	0.08±0.01	**1.70±0.02**	0.04±0.01	0-70	**1.66±0.01**	0.01±0.01	**1.17±0.01**	0.01±0.01
	TTDDEA	-	-	-	-		-	-	-	-
	BO	-	-	-	-		-	-	-	-
	Metrics	FS ↓	SI ↑	FS ↓	SI ↑		FS ↓	SI ↑	FS ↓	SI ↑
	$f(\boldsymbol{x}^*_{\text{OFF}})$	1.73		1.59			1.55		1.08	
	CMA-ES	-	-	-	-		-	-	-	-
40-90	Tri-mentoring	**1.73±0.04**	0.00±0.00	-	-		-	-	-	-
	CBAS	-	-	-	-		-	-	-	-
	ARCOO	**1.73±0.01**	0.00±0.00	**1.59±0.01**	0.00±0.00	0-60	**1.55±0.01**	0.05±0.01	**1.08±0.01**	0.03±0.01
	TTDDEA	-	-	-	-		-	-	-	-
	BO	-	-	-	-		-	-	-	-

Table 9: Overall results in GTOPX unconstrianed scenario.

	Tasks	GTOPX 2		GTOPX 3		GTOPX 4		GTOPX 6	
	Metrics	FS ↓	SI ↑	FS ↓	SI ↑	FS ↓	SI ↑	FS ↓	SI ↑
	$f(\boldsymbol{x}^*_{\text{OFF}})$	275.27		228.84		322.52		142.09	
	CMA-ES	275.27±2.34	0.00±0.00	228.84±3.46	0.00±0.00	322.52±1.07	0.00±0.00	142.09±0.39	0.00±0.00
	Tri-mentoring	251.33±50.42	0.54±0.12	185.64±64.75	0.63±0.05	244.74±87.17	0.41±0.02	135.97±15.99	0.76±0.21
0-60	CBAS	275.27±2.39	0.00±0.00	96.03±10.36	0.84±0.32	275.46±79.07	0.22±0.12	116.67±43.99	0.37±0.26
	ARCOO	**51.97±4.85**	0.91±0.04	129.52±66.78	0.67±0.27	**210.33±103.27**	0.59±0.21	**44.96±8.72**	0.85±0.05
	TTDDEA	959.47±2023.91	−∞	95908.88±129415.48	−∞	11405.36±28334.25	−∞	126.99±60.14	−∞
	BO	93.68±10.56	0.81±0.09	**69.67±14.92**	0.81±0.06	106.5±23.94	0.79±0.11	51.66±5.91	0.73±0.01
	CCDDEA	276.57±4.77	−∞	228.19±8.06	−∞	326.68±5.93	−∞	142.58±2.14	−∞
	$f(\boldsymbol{x}^*_{\text{OFF}})$	218.12		171.54		241.62		121.48	
	CMA-ES	218.12±1.78	0.00±0.00	171.54±0.81	-0.33±0.01	241.62±1.4	0.00±0.00	121.48±0.38	0.00±0.00
	Tri-mentoring	197.63±53.32	-0.32±1.05	166.81±12.51	0.55±0.07	237.26±11.45	0.43±0.09	107.11±21.16	0.09±0.01
0-70	CBAS	218.12±1.78	0.00±0.00	106.53±38.2	0.73±0.12	232.11±19.59	0.16±0.08	83.22±38.49	0.32±0.13
	ARCOO	**54.33±7.18**	0.90±0.03	101.59±35.56	0.73±0.04	190.24±72.22	0.38±0.17	**47.97±7.14**	0.82±0.05
	TTDDEA	227.95±59.71	−∞	28243.65±61226.1	−∞	48026.88±114654.58	−∞	115.87±25.63	-0.89±0.21
	BO	89.69±29.44	0.75±0.14	**76.77±16.35**	0.73±0.08	**108.33±24.68**	0.72±0.16	62.53±5.97	0.67±0.01
	CCDDEA	219.22±3.95	−∞	172.37±5.26	−∞	242.95±4.04	−∞	121.70±1.72	−∞
	$f(\boldsymbol{x}^*_{\text{OFF}})$	175.10		134.83		194.03		102.96	
	CMA-ES	175.10±0.75	0.00±0.00	134.83±0.55	0.00±0.00	194.03±0.89	0.00±0.00	102.96±0.29	0.00±0.00
	Tri-mentoring	118.95±42.5	0.66±0.28	127.93±13.42	0.29±0.04	157.27±22.77	0.69±0.21	94.47±15.1	0.56±0.06
0-80	CBAS	175.10±0.75	0.00±0.00	99.02±20.89	0.66±0.27	183.27±19.71	0.10±0.02	82.24±0.27	0.35±0.01
	ARCOO	**52.34±0.06**	0.87±0.00	77.74±23.06	0.70±0.09	140.88±53.32	0.42±0.20	**46.64±19.93**	0.75±0.27
	TTDDEA	282.21±104.01	−∞	47263.8±103752.22	−∞	274.49±7796.13	−∞	126.69±68.7	-0.45±0.24
	BO	100.94±22.39	0.64±0.14	**70.51±16.03**	0.68±0.12	**114.11±20.68**	0.60±0.19	65.48±11.99	0.60±0.09
	CCDDEA	176.53±3.76	−∞	130.06±15.22	−∞	197.42±10.85	−∞	103.49±1.53	−∞
	$f(\boldsymbol{x}^*_{\text{OFF}})$	136.28		102.69		153.62		83.37	
	CMA-ES	136.28±0.39	0.00±0.00	102.69±0.49	0.00±0.00	153.62±0.6	0.00±0.00	83.37±0.23	0.00±0.00
	Tri-mentoring	111.71±34.31	0.20±0.04	101.23±4.07	0.01±0.00	151.09±6.83	0.17±0.06	83.37±0.23	0.00±0.00
0-90	CBAS	136.28±0.39	0.00±0.00	97.96±6.29	0.07±0.01	153.62±0.60	0.07±0.00	68.54±19.25	0.36±0.12
	ARCOO	**48.89±6.95**	0.83±0.05	**74.05±20.14**	0.28±0.04	129.47±41.1	0.28±0.07	**46.41±14.51**	0.74±0.12
	TTDDEA	2100.55±4408.74	−∞	300661.39±756671.3	−∞	202637.99±534082.56	−∞	101.54±23.1	−∞
	BO	103.95±14.57	0.47±0.02	75.71±15.56	0.45±0.07	**121.59±28.12**	0.45±0.12	62.64±13.96	0.46±0.14
	CCDDEA	138.24±2.59	−∞	491.55±954.25	−∞	153.44±3.53	−∞	83.20±1.05	−∞

Table 10: Overall results in GTOPX constrianed scenario.

	Tasks	GTOPX 1		GTOPX 5		GTOPX 7	
	Metrics	FS ↓	SI ↑	FS ↓	SI ↑	FS ↓	SI ↑
0-60	$f(\boldsymbol{x}^*_{\text{OFF}})$	114.82		-		564.53	
	DE-PF	803.62±1.43	$-\infty$	-	-	4237.06±187.67	$-\infty$
	DE-SPF	805.64±0	$-\infty$	-	-	4502.47±0.00	$-\infty$
	CARCOO	**114.19±0.41**	0.10 ±0.05	-	-	**355.23±151.35**	0.45±0.25
	CCOMs	225.04±167.60	-2.27±0.89	-	-	4722.46±156.71	$-\infty$
0-70	$f(\boldsymbol{x}^*_{\text{OFF}})$	87.13		-		425.37	
	DE-PF	803.62±1.43	$-\infty$	-	-	4237.06±187.67	$-\infty$
	DE-SPF	805.64±0	$-\infty$	-	-	4502.47±0.00	$-\infty$
	CARCOO	**65.97±17.87**	0.25±0.13	-	-	**148.98±14.94**	0.75±0.02
	CCOMs	31.38±0.10	$-\infty$	-	-	4301.71±454.23	$-\infty$
0-80	$f(\boldsymbol{x}^*_{\text{OFF}})$	65.17		-		278.11	
	DE-PF	803.62±1.43	$-\infty$	-	-	4237.06±187.67	$-\infty$
	DE-SPF	**23.72±0.00**	0.94±0.00	-	-	4502.47±0.00	$-\infty$
	CARCOO	64.55±0.41	0.12±0.07	-	-	**153.72±87.96**	0.52±0.34
	CCOMs	602.96±163.72	-7.27±0.45	-	-	4722.46±156.71	$-\infty$
0-90	$f(\boldsymbol{x}^*_{\text{OFF}})$	47.56		-		144.91	
	DE-PF	803.62±1.43	$-\infty$	-	-	4237.06±187.67	$-\infty$
	DE-SPF	805.64±0.00	$-\infty$	-	-	4502.47±0.00	$-\infty$
	CARCOO	**47.77±0.23**	0.19±0.10	-	-	**144.85±83.09**	0.58±0.28
	CCOMs	611.44±157.56	-14.35±0.56	-	-	4722.46±156.71	$-\infty$

Table 11: Overall results in CEC constrianed scenario.

	Tasks	CEC 1		CEC 2		CEC 3		CEC 4		CEC 5	
	Metrics	FS ↓	SI ↑	FS ↓	SI ↑	FS ↓	SI ↑	FS ↓	SI ↑	FS ↓	SI ↑
0-60	$f(\boldsymbol{x}^*_{\text{OFF}})$	-		1.08		2.13		264.16		8.70	
	DE-PF	**6202.03±91.50**	0.00±0.00	2.04±0.00	$-\infty$	3.20±0.00	$-\infty$	**278.25±0.00**	$-\infty$	**42.31±3.82**	$-\infty$
	DE-SPF	**6202.03±91.50**	0.00±0.00	-	-	3.20±0.00	$-\infty$	312.92±18.99	$-\infty$	58.58±0.00	$-\infty$
	CARCOO	-	-	**0.59±0.22**	0.69±0.21	2.79±0.06	2.78±0.15	-	-	-	-
	CCOMs	-	-	83.49±16.18	$-\infty$	**2.23±0.04**	$-\infty$	290.48±4.02	$-\infty$	-	$-\infty$
0-70	$f(\boldsymbol{x}^*_{\text{OFF}})$	-		1.08		2.00		264.16		6.78	
	DE-PF	**6202.03±91.50**	0.00±0.00	1.91±0.12	$-\infty$	**3.20±0.00**	$-\infty$	364.91±0.00	$-\infty$	**44.75±5.41**	$-\infty$
	DE-SPF	**6202.03±91.50**	0.00±0.00	**1.15±0.00**	$-\infty$	4.20±0.00	$-\infty$	364.91±0.00	$-\infty$	58.58±0.00	$-\infty$
	CARCOO	-	-	2.25±0.46	-6.78±3.98	-	-	-	-	-	-
	CCOMs	-	-	395.63±155.21	-5.68±4.40	4.20±0.00	$-\infty$	**265.01±0.32**	58.68±41.95	-	-
0-80	$f(\boldsymbol{x}^*_{\text{OFF}})$	-		1.08		2.00		264.16		4.89	
	DE-PF	**6202.03±91.50**	0.00±0.00	**1.49±0.12**	$-\infty$	4.20±0.00	$-\infty$	**364.91±0.00**	$-\infty$	**40.01±0.99**	$-\infty$
	DE-SPF	**6202.03±91.50**	0.00±0.00	2.04±0.00	$-\infty$	4.20±0.00	0.00±0.00	**364.91±0.00**	$-\infty$	58.58±0.00	$-\infty$
	CARCOO	-	-	-	-	**2.00±0.00**	-0.04±0.02	-	-	-	-
	CCOMs	-	-	180.50±85.37	$-\infty$	4.19±0.00	$-\infty$	-	-	-	-
0-90	$f(\boldsymbol{x}^*_{\text{OFF}})$	-		1.08		2.00		264.16		4.17	
	DE-PF	**6202.03±91.50**	0.00±0.00	**1.77±0.15**	$-\infty$	**4.00±0.18**	$-\infty$	364.91±0.00	$-\infty$	**49.76±4.61**	$-\infty$
	DE-SPF	**6202.03±91.50**	0.00±0.00	2.04±0.00	$-\infty$	4.20±0.00	0.00±0.00	364.91±0.00	$-\infty$	58.58±0.00	$-\infty$
	CARCOO	-	-	2.26±0.03	$-\infty$	-	-	-	-	-	$-\infty$
	CCOMs	-	-	12.04±3.42	1.43±0.13	-	-	**219.87±112.31**	$-\infty$	-	-

data volume, allowing us to explore the impact of varying data scales. The second setting modifies the distribution of the solution space, enabling us to assess how changes in the distribution affect the performance and efficiency of the system.

Table 12: Overall the different size of the offline dataset results in GTOPX unconstrianed scenario. Details are the same as Table 5.

Datasize $f(\boldsymbol{x}^*_{\mathrm{OFF}})$	N=dim*100 197.25		N=dim*300 195.40		N=dim*500 195.66	
Metrics	FS ↓	SI ↑	FS ↓	SI ↑	FS ↓	SI ↑
TT-DDEA	5594.65 ± 13897.52	$-\infty$	194.56 ± 43.91	$-\infty$	424.3 ± 238.99	$-\infty$
ARCOO	99.07 ± 18.99	0.75 ± 0.07	**93.22±23.27**	**0.79±0.03**	**86.68±17.22**	**0.78±0.05**
Tri-mentoring	197.25 ± 3.70	0.00 ± 0.00	195.40 ± 2.62	$-\infty$	179.85 ± 42.05	$-\infty$
CCDDEA	197.25 ± 3.70	0.00 ± 0.00	195.40 ± 2.62	0.00 ± 0.00	167.82 ± 44.25	-0.02 ± 0.11
CBAS	197.25 ± 3.7	0.00 ± 0.00	195.4 ± 2.61	0.00 ± 0.00	195.66 ± 1.78	0.00 ± 0.00
CMAES	179.21 ± 38.49	0.03 ± 0.07	157.41 ± 51.96	0.04 ± 0.06	172.74 ± 40.97	0.04 ± 0.07
BO	**97.26±11.19**	**0.75±0.03**	99.02 ± 25.92	0.72 ± 0.06	104.80 ± 26.16	0.73 ± 0.05

As shown in Table 12, the experimental results demonstrate that when the data volume is small (N=100), the performance of all algorithms is significantly affected. For instance, the FS values of TT-DDEA exhibit considerable fluctuations, highlighting the difficulty of effectively optimizing the objective function in low-data scenarios. Moreover, under small data volume conditions, the SI values frequently approach 0 or $-\infty$, further confirming that insufficient data volume undermines the convergence and stability of the algorithms. We also observed that BO and ARCOO are minimally affected by data volume and are able to find better solutions, demonstrating their exceptional capabilities.

First, we provide a black-box ground-truth oracle objective function for each task. An initial dataset is obtained by uniformly sampling and evaluating the objective function. Datasets of different difficulty levels are constructed based on this sorted initial dataset. Specifically, the right-$n\%$ of the solution space range and the left-$m\%$ of the solution space range are removed to show different solution space distributions. Through the above steps, an offline dataset with a narrow distribution in real tasks is constructed by removing solutions.

To simulate a more realistic data distribution, we choose a dataset size that is 1000 times the variable dimension. At the same time, to further simulate the narrow distribution, missing the $m\%$ and the $n\%$ are used to construct different solution distributions. In this paper, we select the middle 50% of the data (i.e., $m\% - n\% = 50\%$) to construct a simulated dataset as a reasonable baseline. Since the proposed benchmark is highly flexible and customizable, it enables users to modify the data volume, $m\%$, and $n\%$ as needed.

The experimental results in Table 13 show that the FS values of all algorithms are close to the optimal solutions of the offline dataset, indicating that under this experimental setting, it is challenging for the algorithms to further improve upon the current solutions. This phenomenon can be attributed to two key factors:

- Lack of solution space exacerbating the OOD problem: In the experiments, parts of the solution space may not be covered by the offline dataset, which restricts the algorithm's exploration capabilities during the optimization process. As a result, the algorithm struggles to identify potential optimal solutions within the solution space. This limitation often leads to the "OOD" problem, where the algorithm's performance in new optimization tasks is constrained by the incomplete representation of the solution space.

- The offline dataset's optimal solutions are already highly competitive: The offline dataset itself contains solutions that are close to the global optimum. With such high-quality data, the algorithms have limited room for further optimization, making it difficult to achieve solutions that significantly surpass the current optimal results.

Future research could focus on addressing the OOD problem by constructing more diverse and representative datasets or developing algorithms with stronger generalization capabilities to better handle the limitations of incomplete solution spaces.

Table 13: Overall the different solution distribution results in GTOPX unconstrianed scenario. Details are the same as Table 5.

Right-Left $f(x^*_{\text{OFF}})$	0-50 45.67		10-60 40.12		20-70 38.53		25-75 40.62	
Metrics	FS ↓	SI ↑	FS ↓	SI ↑	FS ↓	SI ↑	FS ↓	SI ↑
CMAES	107.89 ± 150.49	$-\infty$	46.92 ± 19.31	$-\infty$	47.41 ± 23.79	$-\infty$	59.78 ± 49.68	$-\infty$
Tri-mentoring	**45.67±4.63**	$-\infty$	**40.12±2.97**	$-\infty$	**38.53±2.48**	$-\infty$	**40.62±1.61**	$-\infty$
CBAS	**45.67±4.63**	0.00 ± 0.00	**40.12±2.97**	0.00 ± 0.00	**38.53±2.48**	0.00 ± 0.00	40.62 ± 1.62	0.00 ± 0.00
ARCOO	**45.67±4.63**	$-\infty$	41.35 ± 3.00	$-\infty$	**38.53±2.48**	$-\infty$	41.63 ± 2.81	$-\infty$
TT-DDEA	56527.15 ± 72621.35	$-\infty$	50225.22 ± 112867.53	$-\infty$	187.73 ± 56.53	$-\infty$	549.99 ± 480.58	$-\infty$
BO	70.88 ± 8.96	$-\infty$	70.98 ± 5.78	$-\infty$	82.18 ± 16.69	$-\infty$	86.16 ± 17.65	$-\infty$
CCDDEA	53.34 ± 21.36	$-\infty$	**40.12±2.97**	0.00 ± 0.00	**38.53±2.48**	0.00 ± 0.00	113.09 ± 146.09	$-\infty$

E HYPERPARAMETER ANALYSIS

We outline guidelines for hyperparameter selection for methods evaluated in the benchmark. These principles help in offline tuning for new tasks. Begin by identifying essential hyperparameters, like learning rate and batch size, and select a tuning method such as Grid Search or Random Search. We run experiments to find the satisfied hyperparameter configurations with performance metrics like accuracy and F1 score.

E.1 HYPERPARAMETERS OF BO-QEI

The primary tunable components of Bayesian optimization include the objective function and the parameters within the optimization loop. The objective function is commonly optimized with a maximum likelihood method, allowing validation log-likelihood or regression error to be tracked directly. Training continues until the validation loss is minimized, ensuring good generalization beyond the training dataset. This tuning method is fully offline since it exclusively uses a static task dataset. In this study, we employ a Gaussian Process model combined with the quasi-Monte Carlo Expected Improvement acquisition function. For an in-depth understanding of various Bayesian optimization strategies and their associated hyperparameters, detailed information can be found in the BoTorch documentation at `https://botorch.org/docs/overview`.

E.2 HYPERPARAMETERS OF CMA-ES

The primary configurable elements of Covariance Matrix Adaptation (CMA) techniques include the trained objective function and the evolution strategy parameters. Typically, the objective function is trained using a maximum likelihood approach to ensure optimal performance. For further details and comprehensive information, refer to the open-source CMA-ES implementation and its extensive documentation available at `https://github.com/CMA-ES/pycma`. In this research, we adopt the default settings for CMA-ES provided by this implementation, with the parameter σ set to 0.5.

E.3 HYPERPARAMETERS OF AUTOFOCUSING CBAS

Autofocusing CbAS has one main tunable components. It involves verifying the performance index, which plays a crucial role in generating the generalization capability of the learning model. By ensuring that the performance index stays above a certain positive threshold, we can ascertain that the algorithm is well-tuned and performing optimally. In this paper, we have set this threshold to 0.9. This careful selection helps maintain the reliability and effectiveness of the autofocusing CbAS approach.

E.4 HYPERPARAMETERS OF ARCOO

The Accumulative Risk Controlled Offline Optimization (ARCOO) method has two hyperparameters: accumulative risk control and Langevin dynamics steps. The former represents the level of total risk tolerance, with an initial momentum set at 0.2, reflecting a moderate risk tolerance. The latter involves sampling by 64 steps of Langevin dynamics, determining the distance between high-risk solutions and observed solutions.

E.5 HYPERPARAMETERS OF TRI-MENTORING

We tuned two critical hyperparameters: the number of neighborhood samples and the number of optimization steps. Due to the algorithm's robustness to the number of neighborhood samples, we set this parameter to 10. To maintain consistency, we standardized the number of optimization steps to 100 across all experiments.

E.6 HYPERPARAMETERS OF TTDDEA

In this evolutionary offline optimization task, we retained the original hyperparameters because they were carefully tuned for mutation rate, crossover probability, and selection pressure. These settings ensure a balanced exploration and exploitation process, preventing premature convergence and maintaining a diverse solution population. Keeping them unchanged guarantees the algorithm's robustness and reliable performance.

E.7 HYPERPARAMETERS OF CARCOO

The hyperparameters in the constrained ARCOO (CARCOO) method need to be consistent with the original ARCOO settings. Therefore, we have configured them to exactly match the hyperparameters used in the original ARCOO method.

E.8 HYPERPARAMETERS OF CCOMS

Constrainted conservative objective models have four main tunable parameters. The first parameter is the degree to which the objective model is allowed to overestimate the objective value for off-manifold designs, set to 2 to ensure a conservative approach. The second parameter is the number of gradient ascent steps, chosen to be 100, balancing optimization efficiency and computational effort. The third parameter is the learning rate, set to $2\sqrt{d}$, where d is the dimension of the design space, ensuring step sizes are appropriately scaled to the problem's complexity. The final parameter is the constraint trade-off coefficient, set to 1, ensuring an equal emphasis on optimizing the objective and satisfying the constraints. These parameters are crucial for fine-tuning the model's performance within the desired constraints.

E.9 HYPERPARAMETERS OF DE-PF & DE-SPF

This hyperparameter is identical to the one used in TTDDEA, maintaining consistency with the original paper. However, due to the algorithm's suboptimal performance in handling out-of-distribution (OOD) problems, we have adjusted the number of iterations to 50 in this study. This modification aims to improve the algorithm's efficiency and effectiveness in addressing the specific challenges posed by OOD scenarios.